Utrecht University

Masterthesis

Support for Resilience in Human-Robot USAR Teams

Author
Jelte B. van Waterschoot

Supervisors
Dr. Jurriaan van Diggelen
Prof. Dr. John-Jules Ch. Meyer

A thesis submitted in fulfilment of the requirements for the degree of MSc in Artificial Intelligence

in the

Department of Information and Computing Science

September 2015

UTRECHT UNIVERSITY

Abstract

Faculty of Science
Department of Information and Computing Science

MSc in Artificial Intelligence

Support for Resilience in Human-Robot USAR Teams

by Jelte B. van Waterschoot

Urban search and rescue (USAR) teams often work in a dangerous, stressful and unpredictable environment. Danger can be partly solved by using robots, as these can reach places too dangerous for men. Humans working with these robots require effective collaboration in performing rescue tasks. In an unpredictable environment like with USAR a good distribution of tasks among the members of a rescue team is of great importance. A dynamic task allocation approach would suit the needs of an USAR team, as members of the team should be quickly reassigned if need be. Workflows have been a favourable approach to model task allocation, yet they often remain static. This research project will look into methods for creating dynamic workflows to support task allocation and provide the means for an effective human-robot collaboration in USAR.

Acknowledgements

At first I want to thank the people at TNO for giving me the opportunity to perform this research internship. I would like to thank my supervisor Jurriaan who was always encouraging new ideas I had and guided me through the challenge of combining multidisciplinary aspects into a good research project. He invested a great deal of time in my supervision for which I am grateful. Also the help from his colleagues made my internship a great success and I have learned a great deal from them.

Next to my supervisors I am grateful for the discussions I had with my co-interns, Serena, Bas and Ramona in particular, the frustrations we shared when we were stuck with an idea, had a programming issue or were uninspired to write. They also made it *gezellig* in the office.

Finally I want to show my gratitude to my university supervisor, John-Jules, who was always enthusiastic about my research project and provided constructive feedback and helped me to achieve the academic level I wanted for this research project.

Contents

Chapter 1

Introduction

Urban search and rescue (USAR) plays a significant role during large-scale disasters, such as the devastating earthquakes in Nepal in spring 2015. An USAR team is responsible for performing rescue operations in these kinds of large-scale disasters and safely rescue the victims from the disaster site. USAR is a necessary – but also a highly dangerous undertaking – and rescue workers need to be supported in every way to ensure their safety. The environment of a search and rescue mission is often chaotic and changes unexpectedly and rapidly. Rescue workers suddenly become trapped under a collapsed floor or a hazardous explosion made the field inaccessible for rescue workers. It would therefore benefit rescue teams to use robots in the field for dangerous missions. USAR human-robot teams can be the solution, where robots can perform the dangerous work and humans can use their expert knowledge and adaptiveness to control the situation. Such a team with humans and robots is an example of a *socio-technical* system, where both humans and technology play their part. The application of robots in the field is yet far from common practice [1]. Only a few cases have been described involving the use of rescue robots [1, p. 1151]. Robots are not as robust as desired and do not always operate smoothly. Taking into account the minimal experience and knowledge of rescue workers operating robots, human-robot teamwork for USAR seems far from trivial. Therefore a support system could contribute to smoothen the integration of and interaction with robots in the USAR domain.

This thesis is part of the TRADR-project, which is the Long-Term Human-Robot Teaming for Robot Assisted Disaster Response project [2]. The TRADR-project addresses five different areas of USAR.

- Persistent environment models
- Persistent models for acting
- Persistent models for multi-Robot Collaboration

- Persistent models for human-Robot Teaming
- Persistent models for distributed joint situation awareness

Persistence involves keeping a common understanding of the disaster area among different USAR teams and their members. Persistence is the key-word for TRADR, which is a great challenge in USAR due to the changing environment, miscommunication and technical failures. Despite these challenges, a human-robot team needs to execute its task efficiently. This research will contribute to creating persistence by the use of a support system in two ways. First, human factors such as situation awareness, how tasks can be supported, and how a support system can be evaluated is discussed. Secondly, a task support system is proposed based on combining the human factors with existing technologies for dealing with unpredictable environments.

The primary task of an urban search and rescue team is to save victims. The USAR mission needs to be executed swiftly, safely and efficiently [3]. First the disaster site requires initial exploration and team members will gather relevant information during exploration tasks. Secondly, an assessment on more detailed scale is provided. The next stage is that any victims found alive need to be extracted from the site as fast as possible, without endangering the rescue workers and victims and without causing additional damage to the site. Meanwhile, any hazardous materials encountered during assessment need to be dealt with as well . Robots can prove themselves valuable assets in USAR, because they require less strict safety measures before entering the disaster zone and could enter places impossible for rescue workers to reach. The SHERPA project is an example of search and rescue disasters in alpine environments and where UAVs support the rescue workers in exploration with unreachable places [4].

According to Murphy et al. there are several tasks during urban search and rescue that could be executed by robots, on which the focus of this thesis project is *reconnaissance* and *structural inspection* [1, p. 1152]. These represent respectively the initial exploration and assessment phase of the rescue process. The corresponding tasks are jointly executed by the human-robot team performing the rescue mission. Murphy concludes that there are two major conditions in USAR which make the human-robot interaction challenging, an unpredictable environment and robots as extensions of humans [3]. The unpredictable, or dynamic, environment poses several challenges, such as keeping all team members up-to-date of the current situation. An example of good information representation would include visualizing the information about rescue processes in a concise and clear manner [5, 6]. The second condition can be a challenge because of the novelty of effective robot use and difficulties of tele-operation [7].

Creating a system to support task distribution that incorporates the two conditions is likely to be beneficial to USAR teams. Due to the complexity of the domain and designing a system suitable for both humans and robots, an appropriate method should be used. The situated Cognitive Engineering method (sCE) is an iterative design method that incorporates not only domain analysis, but human factors and technology as well [8]. The method has proven useful in designing (automated) systems that support humans and is chosen for evaluating the human factors [9, 10].

In particular for our study we look at the task distribution method of USAR missions to support the team leader. Developing a task distribution method has been done before for the Coast Guard's SAR of the Netherlands, which has similar dynamic situations as USAR [11]. Lijnse et al. have used a flexible workflow model approach, and have shown it is well applicable in dynamic situations [11]. A flexible workflow approach offers both the clarity of processes within USAR and fulfills the need to adapt to dynamic situations. Applying flexible workflows in the USAR domain for adding support for task distribution has not been done before and is the core of this research project.

Another addition in USAR are robots as members of the USAR team, which also execute several tasks next to the human team members. Three principles are important in interdependent human-robot teams, namely *observability*, *predictability* and *directability* (OPD-principles) [12, p.68]. The first indicates that relevant knowledge to all members of the team is accessible. Predictability concerns team members being able to anticipate other team members' actions and incorporating this information for planning their own actions. The final one, directability, concerns one's ability to direct the behaviour of others or being directed. These principles serve as guidance for constructing the requirements of the task distribution method. A process-aware information system (PAIS) for executing USAR missions effectively must support the team based on the OPD-principles. A human factors analysis is performed for constructing a workflow model of the USAR domain. From this analysis we can draw requirements and claims. This analysis is used in socio-technical domains and should suit the needs of a human-robot team in USAR. A similar methodology was used in the naval domain for designing a socio-technical system [10].

In our research workflow patterns are added to the support system to model the desired flexibility within USAR [13]. Workflow patterns can offer more flexibility to workflow models, such as support for ad-hoc changes. Another advantage is that the workflow patterns can be directly tested in the workflow modeling phase. The advantage of creating model-based executable workflow models is the fast iterative process for testing and developing [14]. There is no need for a full implementation of the system, it only requires updating the model to the requirements of the USAR team. Rapidly developing a model suits the need of quick functional testing and validating the model.

To paint the picture of possible problems within USAR that could be solved with flexible workflows, a problem example is given below.

> *Problem example* - A team leader has an USAR team consisting of resources (in-field rescuers, robot operators and robots). The team leader in the field has to make decisions about what his team has to do. He uses a walkie-talkie to communicate with his robot operators and in-field rescuers, who have to explore and assess the area and save victims. The team leader can only attend one part of the mission at a time. One of the robots is driving to a location to take a picture, yet at arrival the camera is broken and the operator reports to the team leader no footage can be captured. The team leader needs the information of the site and asks through the walkie-talkie if another robot is available to take pictures. Meanwhile the flying robot has found victims and a team needs to be sent in to extract the victims. The team leader first instructs the in-field rescuers for saving the victims. Precious time is being lost on finding an appropriate resource to take the pictures of the other area. The team leader is experiencing stress and fatigues quickly. However, he continues to coordinate the mission, even though he should be relieved by a team leader of another sortie[1], making him prone to errors. The new sortie has to be updated of current events and be assigned tasks as soon as possible.

The most important stakeholder in this problem example is the team leader, who coordinates the USAR team. He wants his team to operate as autonomously as possible and has to ensure the safety and efficiency of his team. The support system should be easy to use for the team leader, based on knowledge about current standard procedures. The other stakeholders in USAR are the team members, who desire clear instructions, safety and easy access to the system. The remaining actors are the victims and robots. When looking at the problem scenario there are a few requirements the team leader desires. First of all, the team leader wants to keep a good overview of his team members, for example to know if they still operate in a safe environment. Secondly, decisions for coordinating the team must not take up much time and need to be made swiftly and efficiently. Thirdly, the team leader desires to experience less fatigue, thus the stress levels should be reduced. Finally, information transfer between the old and new sortie should go as smoothly as possible.

Four goals have been formulated based on these factors.

- Increase the overall USAR team efficiency in real time, specifically by
    1. making it more efficient for the team leader to oversee the situation
    2. making it more efficient for the team leader to direct his team
    3. preventing the team leader from cognitive fatigue

[1] A sortie is a complete USAR team starting the mission and can be relieved by another team, this team is called the new sortie.

4. making information transfer between sorties more efficient

In order to achieve those goals it is necessary that all activities of the team can be modeled clearly, such that any other USAR team can interpret the model only in a single way. Next to the modeling it is necessary that changes in the model can happen dynamically and that the team leader should be able to adjust his plans accordingly. Lastly, the support system should guide the team leader in where to direct his attention. The system should at least be usable in situations where one has to explore the disaster site, assess the structures on-site and extract victims safely, as this is a requirement of the TRADR scenario [2].

In Figure 1.1 is shown how workflow technology could support task distribution in USAR according to us. At first the workflow technology serves as a basis for generating procedures, which are similar to current USAR protocols. Once the mission is started, these protocols are executed into plans, which consists of processes and tasks assigned to resources. These plans are presented in an appropriate user interface. Whenever the team leader needs to give new directives or the environment changes, he uses the interface to update the plans which will adjust procedures and adapt the flow in the workflow engine. These ideas from workflow technology serve as the basis for designing a support system.

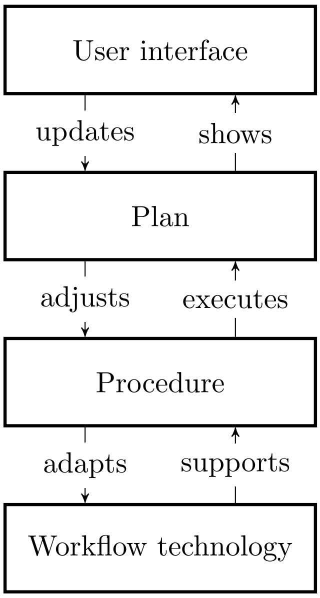


Figure 1.1: The benefit of a system based on workflow technology is its adaptation to the situation via procedures and plans and interacting with the user interface.

The human factors of task distribution and workflow technology should help design a good support system for which the following question is asked.

- How can a process-aware information system support identifying and resolving sudden issues in the task execution of an urban search and rescue team?

The main question is two-fold, as there are two subquestions which can support the solution to the main question. The first is a human-factors question, the second about the technological realization.

- How can factors from a human perspective be incorporated for a support system for human-robot USAR missions effectively?
- How can workflow systems provide a more adaptive task distribution among humans and robots which is needed to cope with the unpredictable nature of USAR missions?

The aim of the study is to create a suitable task support system for USAR. The sCE-method will be applied to discovering the human factors relevant for the system and establishing the requirements for the system. Furthermore, an implementation of workflow models for the support system is provided to evaluate the use-cases. A pilot study with end-user will evaluate both the human factors concerning the support system as well as the technical capabilities of the support system.

The thesis continues with Chapter 2 describing the background knowledge about the sCE-method, the human factors relevant to USAR and the envisioned technology to create the support system. Chapter 3 discusses the topics relevant to the human factors and grounds the human factors research question. In Chapter 4 the implementation of the prototype support system is discussed, relevant for answering the technical research question. Chapter 5 contains the evaluation results of the support system, first for the human factors aspect, secondly for the technical aspect. Conclusions and future work are discussed in Chapter 6.

Chapter 2

Background

This chapter starts with a section containing background information about the situated Cognitive Engineering method. It continues with introducing the relevant human factors for the system design, which is relevant for the first research question. The third part contains information regarding the envisioned technology, which is relevant for the second research question.

2.1 Situated Cognitive Engineering

The situated Cognitive Engineering method is developed for designing socio-technical applications, which are (autonomous) systems built for humans to work with. Looije and Neerincx have divided the sCE-method in five components, making it easier to apply the method in developing socio-technical systems (Figure 2.1) [15]. The method starts off with a thorough theoretical foundation relevant to the design of a socio-technical system, the derivation component. This includes the demands of the stakeholders (operational goals), task analysis and capabilities of the stakeholders (human factors knowledge), current autonomous systems (technology). In the second component, specification, requirements based on the foundation help to propose negative and positive claims which are deemed applicable and plausible. Use-cases are situations (hence the situated in sCE) describing the use for the system and help to validate the claims. The third component, the build, is where the system is implemented, based on the requirements of the specification and is often referred to as the prototype system. The evaluation component tests the claims made with respect to the implemented system by using human in the loop testing. It means that not all parts of the build are complete, yet some parts can be tested already with end-users. The fifth component involves a refinement of all other components, and serves as the iterative aspect of the sCE-method, such as adding functionalities to the build or change requirements.

Compared to classical cognitive engineering methods, the sCE-method includes aside from domain and task analysis, knowledge from both human factors and technologies. Including human factors explicitly helps to incorporate requirements for the system, as well as the experimental results of the evaluation. The benefit of the technological input is two-fold, it supports to specify elements precisely and generate feasible ideas. Moreover, the technological effects of the system are made explicit and are already integrated in the design more formally, making development of the system easier from the start. Because the sCE-method encapsulates information on many different levels, it is more flexible in designing a system in which not all of the technological effects are clear. An advantage of the sCE-method is that it can be used in complex domains, as often in those domains not all effects of a system design are evident. Hence, for the complex domain of urban search and rescue the sCE-method is applicable.

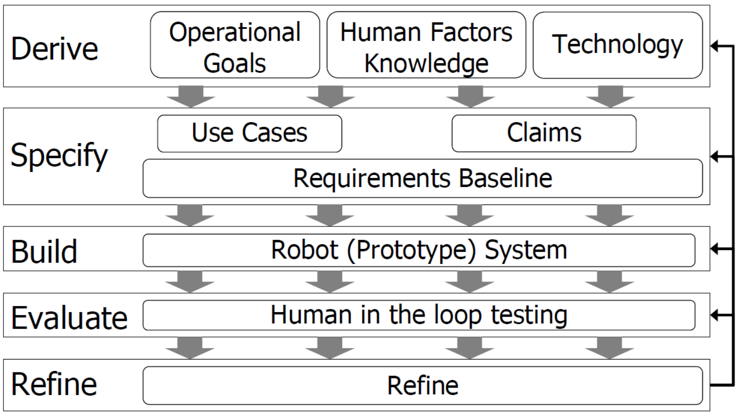


FIGURE 2.1: The situated Cognitive Engineering approach, displaying the relations between the five different components [15, p. 2].

2.2 Human Factors

Two major human factors are the backbone of this research. First, collaboration between team members, human or robot, is discussed. It is important to know how team members can perform multiple tasks synchronously or asynchronously, either being a human or a robot. The second factor is the work analysis of USAR. Cognitive work analysis (CWA) plays a large part in analyzing the processes of USAR. The method can be used to effectively look at the way team members of USAR work in a dynamic environment, such as how decisions are made and how members cope with problems.

2.2.1 Collaboration

Effective collaboration is key to a good human-robot team. According to Murphy, knowledge representation is important for having a shared mental model between all team members [3, p. 151]. One study involving collaboration in a team showed that monitoring the bigger overview is seen as sharing collaborative information, accessible to all members and made collaboration more effective [16, p. 89]. Displaying such information in an efficient way has been proposed with the use of organigraphs [17]. Organigraphs are a form of visualizing collaboration awareness. Van Aart and Oomes developed a program in which different aspects, such as coordination and communication, are displayed graphically in a clear structure [17]. Whenever a problem occurs, it can easily be traced back in the organigraphs and can be resolved quickly. Situation awareness (SA) is closely related to collaboration awareness and a common definition is "[t]he perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future"' [18, p. 36]. Applying the definition in the USAR domain would mean that SA is knowing what each team member is doing, how long the members' actions take, which environmental changes occur, and how this will affect the progress of the mission, closely related to OPD-principles mentioned in the introduction [12]. Another study looked into SA in human robot teaming and found that the human members often have trouble understanding the robots capabilities or roles [7]. Interfaces frequently give incomplete or incorrect information about the situation. Also a problem was that multiple humans were in charge of one robot at the same time, which resulted not only in a poor collaboration between the humans and the robots, but also between the humans themselves.

Challenges from the human-robot teaming in the USAR domain also apply to other dynamic situations, such as aerospace missions with robotic assistants. Smets et al. have tested the emotional response of humans working with robotic assistants during the MARS-500 experiment [9]. The assistants were found useful by the astronauts and with some improvements would make excellent companions during aerospace missions. Another related field mentioned in the introduction is the naval domain, where battleships work with complex automated systems and the commander of the vessel has to keep an overview of the situation, especially during an emergency (e.g. torpedo attack). Van Diggelen et al. created a prototyping test environment, simulating situations on the battleship for quick testing of the automated systems and discovering improvements for the vessel [14]. Not only improving the technology on the human side has been a contribution, some research has focused on improving the robot-side of the team. One way is making it easier to delegate robots and let them operate as autonomous agents, similar to coordinating human team members. A formal framework is proposed to make it easier to delegate robots and has been tested with UAVs during emergency services [19]. However, another study found that treating autonomous robots the same as human team members was actually

[1] https://en.wikipedia.org/wiki/1996_Belgian_Air_Force_Hercules_accident

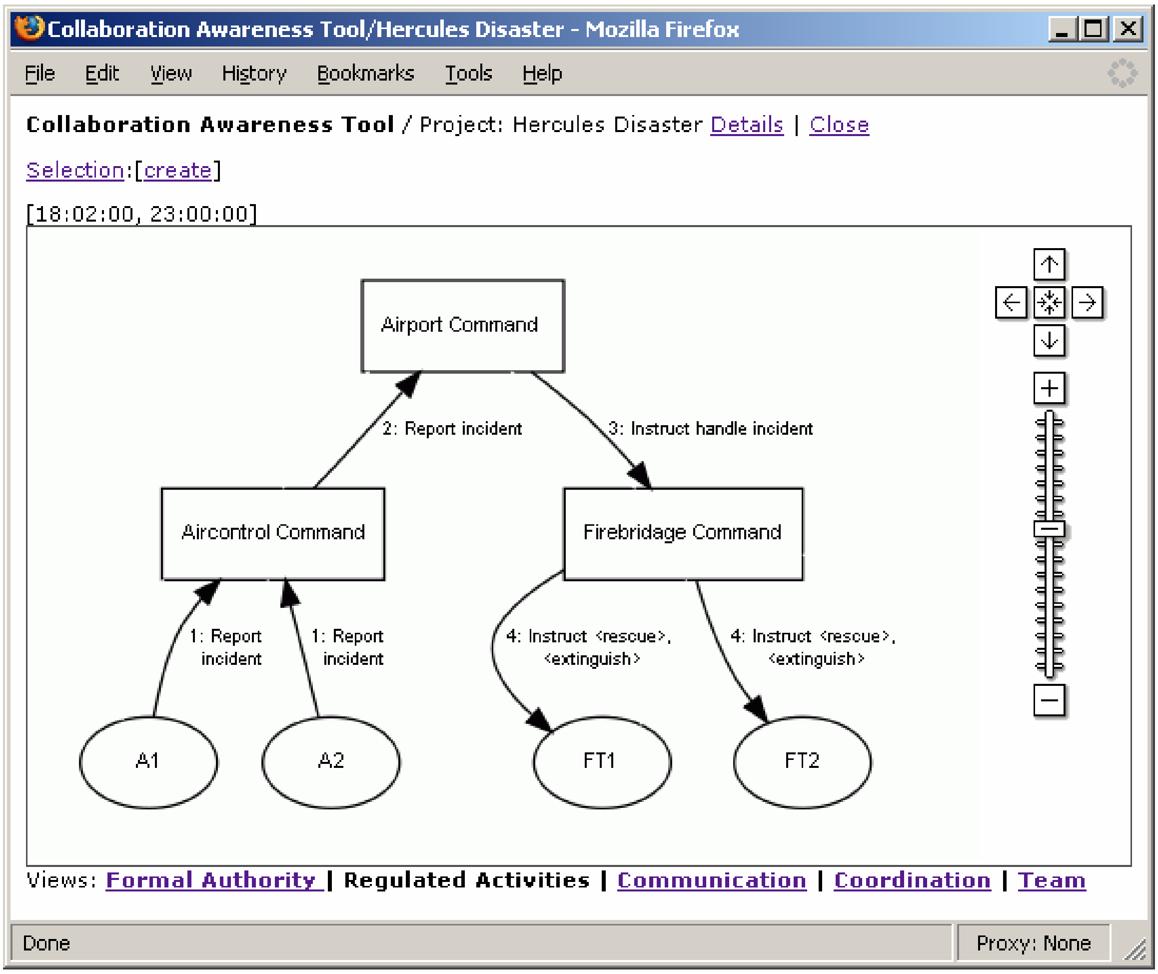


FIGURE 2.2: Abstracted Regulated Activities organigraph of the Hercules Disaster[1] [17].

less effective than when these robots were semi-autonomous and partially controlled by human team members [20]. A possible explanation would be that robots which are capable of more complex actions are harder to create a shared mental model of. In semi-autonomous robots, humans have some idea how the robots perform actions, because humans themselves have to partially control the robots. Fully autonomous robots however work more as a black box and can be harder to create a shared mental model of for the human team members, resulting in a worse situated awareness and thus a weaker team performance.

Decision making is another significant aspect of USAR missions and it is essential that the team leader can make the correct calls. The team leader needs to be aware of changes in the situations of all team members at all times, something which remains often hard for robotic members [1]. To coordinate the mission, the team leader can have different scopes on the rescue process. For example he could watch closely the extraction of a victim, or he can go to the top-level, monitoring all ongoing rescue procedures. When during a mission everything goes according to plan, the decision making is fairly easy. All team members can follow protocol and only few decisions need to be made to run the rescue operation as efficient as possible. However, problems such as communication failures or information missing from certain team members require immediate attention. These problems require the attention of the team leader who has to set priorities and decide what needs to change in the mission process to still successfully complete the mission. He also has to inform the involved team members of changes.

2.2.2 Cognitive Work Analysis

Cognitive work analysis has been a common standard in recent decades for analyzing and representing processes in which both (automated) systems and humans play their part [21, 22]. CWA consists of five types of analyses, of which work domain analysis (WDA) is most commonly used. "WDA is used to define the task environment; the environment that the activity is performed in" [22, p. 18]. As the system should support task distribution, WDA is a good choice. Several distinct models exist to perform a work domain analysis, such as multi-level flow modeling, decision ladder and abstraction hierarchy [23]. These models can be transformed and serve as a basis for designing an effective interface of the system for humans. Multi-level flow modeling (MFM) is focused on work domain goals. MFM requires functional descriptions for achieving the work goals. It is useful when goals change quickly, because of its non-hierarchical structure and event dependencies. When a goal changes, not the entire model should be processed for adaptation. Decision ladders (DLs) model cognitive activities (information processing or decision making), states, and actions that are structurally related to tasks. An abstraction hierarchy (AH) is useful for modeling the work domain structure. AH is focused on describing the purpose of work and often gives a global view of the work domain [22, p. 24 - 26]. The abstraction hierarchy seems to be the most relevant choice for this research, as it has also been applied in similar domains with dynamic environments. Van Diggelen et al. have created an abstraction hierarchy as basis for describing the processes in the naval domain, which mapped correctly on the existing naval system [14].

2.2.2.1 Abstraction Hierarchy

The abstraction hierarchy has two axes on which a system is defined (Figure 2.3). The horizontal axis describes a system based on decomposition, from a very global perspective of the system as a whole until the component level. The vertical axis describes the system from abstract to concrete roles. Especially the second axis is interesting, because people using interfaces based on the AH monitor tasks often on the functional level (third level, called *functions*), but solve problems at the physical objects level of abstraction (bottom level, called *resources*) [24]. The process level (fourth level, called *processes*) describes which physical objects of the bottom level should be used in a process to ensure it on the functional level. Functions are often operational during the total execution of the functional purpose (top-level), whereas processes can be initialized and completed multiple times for ensuring the functions. The values and measures level (second level) is for quantifying how well the functional purpose has been reached, for example if all victims are saved and if it is within a certain timeline. Interesting is that people can reason on the functional level, which is a more abstract function, but the system only replies in a bottom-up way. People are required to switch between different abstraction levels to control the system. The switching between the different levels of abstraction is referred to as people "understanding" the problem that needs to be solved [25].

Two conditions need to be met for designing a good abstraction hierarchy. The relevant information must be present in the AH and it must be correctly represented. Mostly the former can only be done by the designer based on experience and knowledge. The second condition is more of a modelers problem, for whom the job is to construct a suitable formal method [21]. A useful approach to construct the abstraction hierarchy, is by using scenarios and use-cases, which clearly describe the elements on different levels of the AH [26]. Once the AH has been constructed, an appropriate interface can be built.

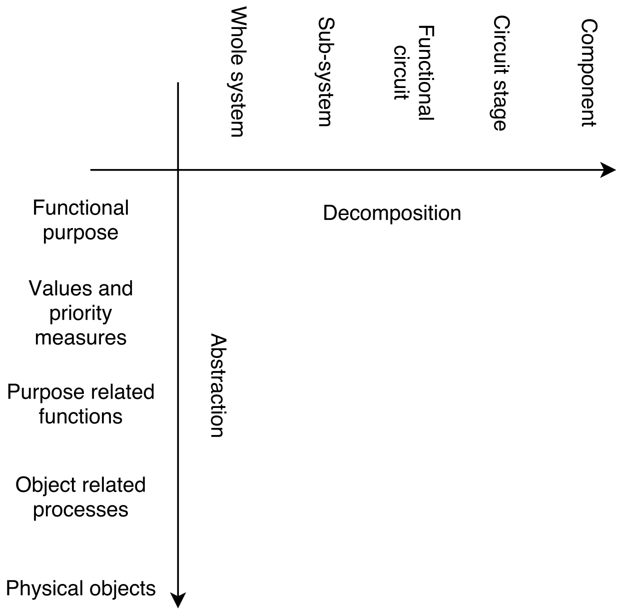


FIGURE 2.3: The two axes of the abstraction hierarchy [21, p. 235].

2.2.2.2 Ecological Interface Design

Ecological interface design (EID) has been first described by Rasmussen and Vicente and has been a familiar design method in designing interfaces for system control [25, 28]. EID is a design methodology based on CWA and takes the task and environment as point of interest for designing socio-technical systems, instead of only the user as in traditional system design. EID is based on differentiating three levels of control, namely skill-based, rule-based and knowledge-based (SRK) [25]. Skills are what most people use everyday, they are familiar with certain procedures and can execute them with little cognitive processing. Rules are used during unfamiliar events, but procedures exist to react to them. Knowledge-based control occurs when people have to improvise when dealing with unexpected events. Whereas skill-based and rule-based are more low-level controlled in the AH, knowledge-based reasoning occurs on a higher level.

An EID interface supports the switching between the three levels smoothly and increases the efficiency of the user interacting with the system. Sam has created an EID for collaborative partners during a crisis situation (Figure 2.4) [6]. Another study has also confirmed the effectiveness of an EID, by comparing a standard physical interface with an interface which shows information from different levels of the AH [29].

2.3 Envisioned Technology

The technology should be device-independent, because the system should be usable on both stationary and mobile devices. Options include a browser-based system or an application for both mobile and stationary devices. The system should be web-based such that it is easier accessible from any device. The in-field rescuers would be provided with only a smartphone or small tablet, which has the advantage of being portable, yet powerful enough to see relevant information. Laptops and larger tablets however have a larger screen and are capable of displaying more information, which suits the needs of the team leader who has to keep an overview. In addition, EID has also been proven to be useful for smaller devices as well as stationary ones [30]. For a further overview of all technologies required for a support system, see Table 2.1.

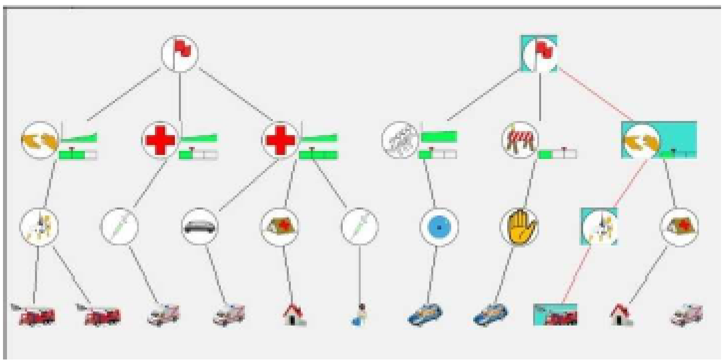


FIGURE 2.4: An example of an ecological interface design for crisis management [6, p. 32]. You can see the AH structure in the interface. The visualization shows how the partners are related in performing certain functions and how well they are doing (the green bar indicator). The highlighted AH on the right shows how a function and purpose are related to a particular physical object.

Technology	Benefits	Downsides
Tablets for mobile team members	Easy wearable, quick access	Prone to network failure, prone to damage
PCs for stationary team members	More information, stable connection	Low mobility
Browser-based solution	Universal and easy access	Prone to network failure, possible hacking
Server-based solution for logging processes	Central storage and easy access	Network failure could cause data loss, possible hacking
Workflow-based representation	Visually clear, maps processes well	Too abstract, no integration in TRADR system

Table 2.1: Technologies required for a support system.

2.3.1 Flexible Workflows

A workflow is "[t]he automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules," defined by the Workflow Management Coalition [31]. Workflows were primarily designed after people have manually constructed how to organize their businesses [32]. Today however, workflows are constructed beforehand to organize a business, which is cost-efficient and more structured. These workflow models are often called Business Process Models (BPMs). An example of a BPM would be the ordering process at any warehouse. In this case, the process is divided into different tasks, such as processing an order and contacting the supplier, which are all elements of the process model. Modeling the processes in such a way gives clear insight into who is involved with the process, what resources they require, and what the order of tasks is. Many businesses can be optimized by using workflows, for example to reduce workload on people [32, p. 7].

Workflows are mostly intended for business processes, which are often static. There have been few attempts to apply workflows to the domain of USAR. One study involved describing the actions and tasks of robot use during search and rescue, though only focuses on specific single-robot uses [5]. Some workflows do represent multiple robots in an USAR team, but only represent it statically [3]. The current study takes it one step further and will look at how all of the tasks involved in a rescue mission can be coordinated dynamically. By adopting the model shown in Figure 1.1, we hope not to only develop a suitable support system for USAR, but contribute to the overall use of flexibility in crisis management and other domains that call for flexibility.

Different modeling languages exist for creating BPMs, such as BPEL and BPMN [33, p. 5-7]. BPMN is a popular choice in industry nowadays and has been extensively developed. The format is fairly easy to use interchangeably with different BPMN developer environments. BPMN stands for Business Process Model and Notation and is often applied to develop BPMs. The advantage of BPMN is that it is both expressive enough for engineers to model processes correctly and still comprehensible for less tech-savvy people improving the business processes. BPMN has formal semantics for modeling BPMs, which makes it less ambiguous than other modeling languages [34]. Another bonus is case handling, which means that complex processes with different process flows can be modeled, making the workflow model suitable for dynamic processes. Recent developments added the possibility to run BPMs as web-services, making them easier accessible [35]. BPMNs are also capable of running device-independent, such that any user of the system could switch device in real-time.

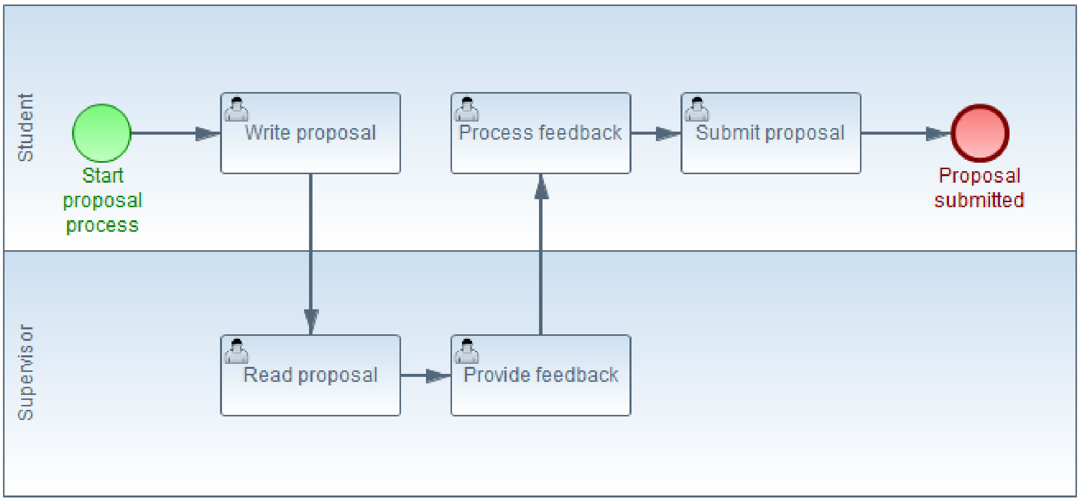


FIGURE 2.5: A process model describing a simplified procedure for submitting a research proposal.

See Figure 2.5 for a workflow modeled in BPMN, which will serve as an example to elaborate on the workings of workflows. A workflow model consists of a single starting event, at least one task and at least one ending event. The green circle indicates the starting event, representing the trigger for starting the process. The trigger is not clearly stated in workflow literature [13, 32]. Therefore the assumption in this research is that a trigger of a process is a human or script, specifically for USAR it is the team leader. For the workflow model in Figure 2.5 this could be a student who wants to submit a research proposal for his master thesis. The tasks are represented in boxes, containing text which describe the activity necessary for the task. The first task the student has to do, is to write an initial proposal. For each task, information can be entered in the workflow model. The information is stored in *variables* and can be either assigned or updated by the tasks inside the workflow model. Best practice in workflow modeling is that the name of a task always contains a verb [33]. The end event is represented by a red circle, described as the state in which the goal of the process is achieved, namely to successfully submit a proposal[2]. The workflow model in Figure 2.5 has also two big horizontal boxes, which

[2]End events do not necessarily represent a succesfull end state. When there are multiple end events, one states achieving the goal succesfully, whereas the others describe states where the goal is (partially) not achieved

are called *swimlanes*. Swimlanes contain the task per resource, which are a student in the top lane and a supervisor in the bottom lane. At one glance you can see which resource performs which task. The arrows between the tasks dictate the flow of the tasks and can cross lane boundaries any number of times. Tasks can only be executed in the order of the flow described by the process model, for example the student cannot submit a final proposal before receiving feedback. The student example is fairly simple and often workflow models contain more complex elements, such as logical gates or other types of events. In Figure 2.2 the workflow elements will be shown that are relevant for this project, a more extensive overview can be found in the book by Silver [33]. During the execution of workflows, information is passed on from one task to another task within a process. Often information within a process is required in other processes as well. If a hierarchical structure is present, a lower level process can pass information to a higher level process, via sub/super process relations (Figure 2.8). The advantage is that the hierarchical structure clearly indicates on which level information passing currently happens and the details on the low level do not obscure the higher level process. Whenever the information is also relevant for other processes on the same level and there is no hierarchical structure binding them, the information must be sent via *signals*. Signal-linked processes are more difficult to view visually, because there is no clear ordering, which is the case with sub/super process relations.

When taking the abstraction hierarchy into account, the bottom three levels of abstraction (Figure 2.3) can be modeled directly into workflow models. The physical objects level represent the resources, which are indicated by the names in the swimlanes executing the tasks. The object-related processes are all the tasks combined in a process, so the entire process model, such as the process in Figure 2.5. The purpose related function is a little harder to represent, but would be a function such as 'Ensure students graduate'. This function would then have the process of Figure 2.5 to fulfill the function (partially).

Schonenberg et al. have provided a method for developing flexible processes and state that "[p]rocess flexibility can be seen as the ability to deal with both foreseen and unforeseen changes, by varying or adapting those parts of the business process that are affected by them, whilst retaining the essential format of those parts that are not impacted by the variations" [13, p. 2]. A related domain to USAR that calls for flexibility is that of medical care. Patients require personalized help, though most health systems work with a general procedure. Combining workflow technology with an extensive medical ontology, a personalized health plan can be produced for each patient [36]. Another medicinal study has used workflow methods for treatment processes in the hospital. The result was that the flexible workflow approach fitted the dynamic demands of the healthcare domain well [37]. For the USAR domain it is essential to know what impact a sudden collapse of a building or minor events like the discovery of small odd objects has on the ongoing processes of the team. Keeping most processes the same puts the least amount of

Type	Element	Event	Icon	Details
Logical gate	XOR	-		One outgoing connection is chosen on a condition for the continuing flow
	AND	-		All outgoing connections are activated
	INC	-		At least one outgoing connection is chosen, possibly based on a condition
Activity	Script	-		A task consisting of programming code to automatically adjust the workflow model
	Call	-		A superprocess that calls a subprocess. The subprocess is often indicated with a '+' sign on the superprocess
Signals	Throw	Start		A starting event waiting for a particular signal to start the process
		Intermediate		An intermediate event waiting for a signal before continuing flow
	Catch	End		An ending event sending out a signal when the process is completed
		Intermediate		An intermediate event sending out a signal to another process

Table 2.2: Partial list of BPMN 2.0 elements.

pressure on coordination, yet the unexpected events should be appropriately attended. A taxonomy put forward by Schonenberg et al. helps with deciding which flexibility approach suffices for each domain [13]. Their taxonomy includes four types of flexibility.

- Flexibility by Design
- Flexibility by Deviation
- Flexibility by Underspecification
- Flexibility by Change

For each of these types of flexibility we will discuss whether they are suitable for the USAR domain. Evaluation criteria include possibility of use, easiness of use and the relation to the OPD-principles [12]. A leading example for a dynamic context will be a fire interfering with saving victims. An overview of all types can be found in Table 2.3.

Flexibility by Design

The most simple form of flexibility is where all flexibility is modeled beforehand. A normal sequence of executing tasks for USAR is modeled, but also the scenario of what to do when a fire suddenly occurs is described explicitly. An example is shown in Figure 2.6, where a XOR-gate is used as alternative pathway if a fire is present. Flexibility by design thus offers good observability and predictability options, because any current process is modeled explicitly and is easy to predict following actions. Events like fires and sudden collapses can be modeled explicitly in the workflow, yet it seems likely that not all sudden events will be captured upfront. Therefore flexibility by design is sufficient for the normal workflow and common anomalies, but unexpected events will remain outside the scope of the model. As these unexpected events could still be relevant to the team, the events should be encapsulated by the model. Unfortunately that is not possible with a process model based on flexibility by design alone, leaving no good options for directability.

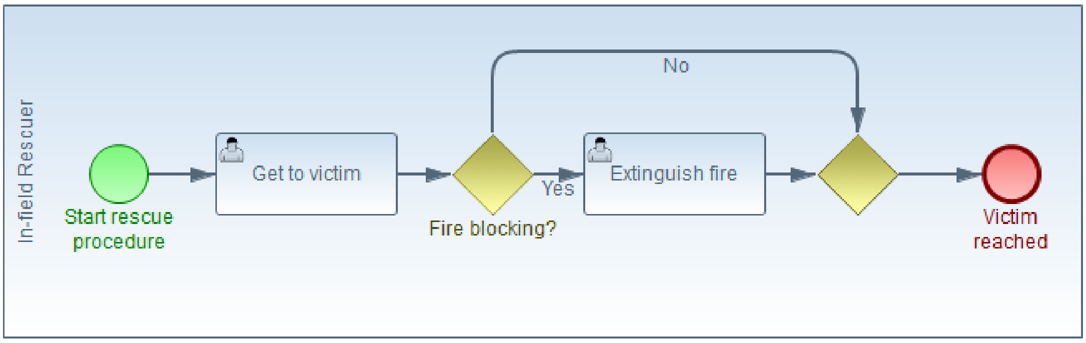
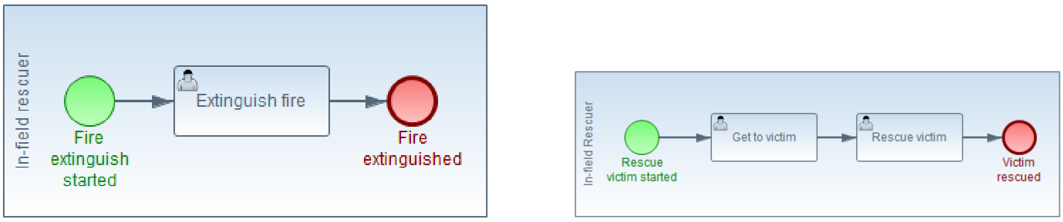


Figure 2.6: An alternative pathway for dealing with fire by using flexibility by design.

Flexibility by Deviation

The second form of flexibility is by deviation. As before with flexibility by design, the process model remains the same during runtime. However, after executing the process model, the order of the tasks and processes could change. Parameters in the process model could dictate the order and could be changed by the team leader or influenced by the environment. An example would be what to do when a fire has occurred (Figures 2.7a and 2.7b). The team leader can choose to first send in an extinguishing party and save the victims afterwards or vice versa. An advantage of flexibility by deviation is that it offers more flexibility than flexibility by design, because it lets the team leader dictate the order of the processes. However, processes and tasks as defined in the process models are static and do not give enough flexibility for the team leader if a problem needs to be solved on the inner-process level. Flexibility by deviation does offer clear process models for observability, but only average on predictability because the order of processes could change. Directability is very limited to deciding only the order of processes.



(A) The process of extinguishing fire.

(B) The process of rescueing victims.

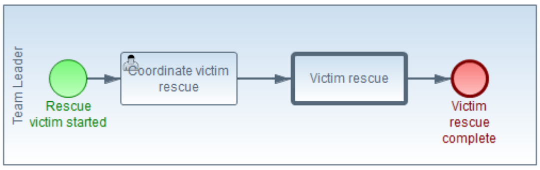
FIGURE 2.7: The team leader has to coordinate the victim rescue and can decide to first extinguish the fire or first save the victims, depending on the situation.

Flexibility by Underspecification

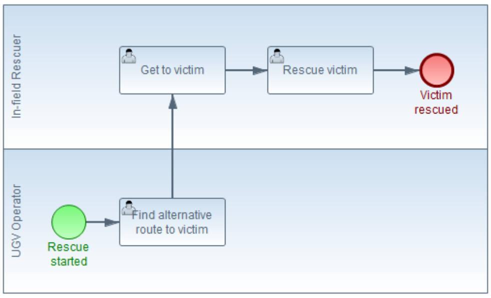
The third form of flexibility is by underspecification. As the name suggests, process models are underspecified, meaning that some information is missing to complete the process at the start of runtime. Once the process has started, the underspecified parts are filled in with complete process models, which could be based on information given by the team leader or environmental influences. In the example displayed in Figure 2.8, the general process of victim rescue is shown with two possible specifications at the bottom. The rescue team can either find an alternative route for the victims (2.8b) or try to extinguish the fire and rescue the victims (2.8c). The advantage of this type of flexibility is that information gathered in the field can dynamically alter the processes. Still, it is necessary to know exactly where these specifications occur, which might not always be the case in USAR. You might make the entire process underspecified, but it would render the comprehensibility of the workflows useless. For observability, flexibility by underspecification would suffice, unless too many parts are underspecified. Predictability-wise, it is on par with flexibility by deviation, due to that complete process models are observable and you can see the flow clearly, but could miss some important details. The directability aspect is better than the two flexibility types before, because information can alter some specifications of tasks.

Flexibility by Change

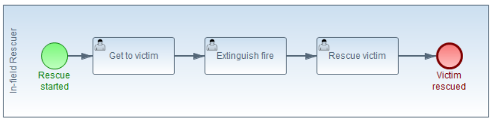
The final form of flexibility is by change, the only form of flexibility where a process model can be changed at runtime. Causes for change at runtime are sudden events, like smoke obscuring the view or a collapse in a building. In Figures 2.9a and 2.9b two process models are displayed. The first is the standard model for rescuing the victims, the second is a modified model at runtime where the in-field rescuers require to find an alternative path because of a blocked exit. Another task is added to react to the collapse event in the second model. The advantage of change flexibility is the possibility to react to any event occurring, even the most rare events. A necessary condition however is that someone with knowledge-based expertise is available and can efficiently change the process models. Flexibility by change is also the most difficult to



(A) A victim rescue which can consist of other subprocesses.

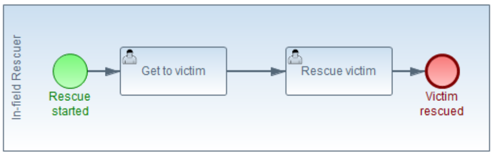


(B) The process of finding alternative route to get to the victim.

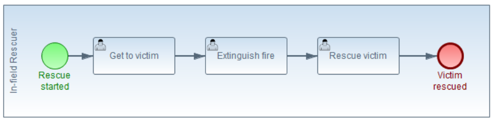


(C) The process of extinguishing a fire before rescuing the victim.

FIGURE 2.8: An example of a process containing a superprocess and two different processes as a possible subprocess.



(A) The first process model of saving victims from a building.



(B) An alternative model is constructed and executed to save the victims.

FIGURE 2.9: An example of how a process model can be adjusted by adding a task in runtime, when a sudden event occurs.

interpret, as the process models can change at any time and do not guarantee consistency. It can become more difficult for team members to have a consistent shared mental model about the situation, rendering the observability factor lower, though all changes are visible. Predictability is hardly present, as process models can change at runtime easily. The strong factor of flexibility by change is directability, because any change is possible on any level of the AH.

In conclusion, as can be seen in Table 2.3, none of the types of flexibility is good on its own. It is not uncommon to use different types of flexibility simultaneously [32]. The best idea would be to pick the best type for the choice of observability, predictability or directability.

2.3.2 Socio-technical Simulation Tool

A support tool has been developed that could incorporate both the OPD-principles and some of the workflow flexibility, called the Socio-technical Simulation Tool (ST2). The first version of ST2 has only recently been developed for supporting task distribution in the naval domain

Principle/Type	Design	Deviation	Underspecification	Change
Observability	✓	✓	o	o
Predictability	✓	o	o	–
Directability	–	–	o	✓

TABLE 2.3: Overview of the OPD-principles of workflow flexibility types. ✓means the flexibility type supports the principle, o means it has some support for the principle and – means it has little or no support for the principle.

[14]. "ST2 is designed to support rapid prototyping of multi-user test applications in the navy domain" [14, p. 6]. It is a task-oriented prototyping tool, as an extra layer on top of an existing programming environment, called iTasks [38]. ST2 helps to support the crew of a navy vessel by increasing the efficiency of monitoring situations. Important to note is the prototyping aspect of ST2, which is already present in the modeling phase of developing a system. Most USAR scenarios can be modeled beforehand and during the modeling phase the system can be evaluated by keeping the end-users in the loop. A major advantage is that it saves much time and effort from implementing an entire support system first (Figure 2.10) before evaluating. In order to be a proficient prototyping tool, ST2 contains the following five features.

- Scenario control
- Generic decision ladder structure
- Dynamic task allocation
- Reactive agents for testing
- Intelligent agents for testing

All these features are relevant to the USAR domain as well. Scenario control encapsulates the possibility to simulate a scenario, such as rescuing a victim. The generic decision ladder structure is concerned with the hierarchical structure, based on SRK (Section 2.2.2.2). The third feature, dynamic task allocation, means that tasks assigned to team members can change or be re-allocated to other team members in runtime. The last two features are specifically convenient for testing purposes, because the tool requires few human testers and can simulate all other required team members. The agents can also be used as simulators of the UAVs and UGVs.

The new version of ST2 is not based on iTasks anymore, but on the JBoss programming environment and jBPM, a Java-based version of modeling BPMN 2.0[3]. It has several advantages over the first version of ST2. One of them is that ST2 runs online, because it is a web-based service. Any device with a web browser can access it. Another advantage is that it becomes easier

[3]Description and documentation of jBPM, http://docs.jboss.org/jbpm/v6.0/userguide/

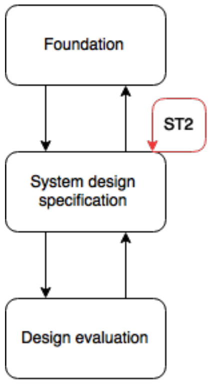


FIGURE 2.10: The situated Cognitive Engineering approach [8], adapted to mark the addition of the ST2 tool.

to build a task distribution support system based on workflows, because modeling workflows visually with minimal programming removes the need for experienced programmers. The new ST2 also supports all components shown in Figure 1.1. USAR protocols, which are often used to execute a successful rescue mission, consist of a sequence of tasks and map good to the way of modeling workflows. These workflows can immediately be executed in runtime. The other benefit is that these plans can be changed dynamically, in runtime. A USAR team may deviate from the plans and update their current plan to match better with the current conditions of the environment. The support system will be automatically be updated with these changes.

2.3.3 Workflow Patterns

Based on the mentioned four flexibility types, several workflow patterns can be distinguished, which are shown in the first column of Table 2.4 [13]. The table also shows how ST2 as a tool could incorporate several workflow patterns that are found necessary in the USAR domain. In section 4.1.2 an analysis is applied to all workflow patterns and a selection is made which will be used for creating and evaluating the support system.

Patterns	Type	ST2?	Required?	Example
Parallelism	F1	✓	✓	AND +INC gate
Choice	F1	✓	✓	XOR + XOR/INC gate
Iteration	F1	✓	✓	XOR + INC gate
Interleaving	F1	✓	–	Double parallelism
Multiple instances	F1	✓	o	Multi-instance process
Cancellation	F1	o	✓	Button opposite of 'perform'
Undo task	F2	–	–	–
Redo task	F2	o	✓	Generate forms for missing data
Skip task	F2	o	✓	Automatically fill in task
Create additional instance	F2	✓	o	Iteration + event signals
Invoke task	F2	✓	✓	Event signals
Late binding	F3	✓	✓	Variable for subprocess binding
Late modeling	F3	–	✓	–
Momentary change	F4 (E)	✓	✓	Resource reallocation
Evolutionary change	F4 (E)	–	✓	–
Entry time	F4 (Mo)	–	✓	–
On-the-fly	F4 (Mo)	–	o	–
Forward recovery	F4 (Mi)	–	o	–
Backward recovery	F4 (Mi)	–	o	–
Proceed	F4 (Mi)	–	o	–
Transfer	F4 (Mi)	–	✓	–

Table 2.4: Overview of all flexibility types. The first column states the workflow pattern, the second column the type of flexibility (F1: Design, F2: Deviation, F3: Underspecification, F4: Change), the third column states if it is feasible in ST2, the fourth column indicates if it's required for the USAR environment and the last column gives an example if it (might) be possible in ST2. Adapted from Schonenberg et al.[13].

Chapter 3

Design Specification

The specification, or, the system design specification, describes how the goals of the system can be translated into a system design that solves the identified problem. The goal of the system is to support an USAR team and specifically the team leader. The support system will be called the TRADR Task Support System (TTSS). In this chapter the human factors research question will be addressed and help to create a design specification. The chapter starts with providing a picture with a scenario and use-cases. Requirements for TTSS are discussed in the following section. The chapter concludes with claims about the specification of TTSS and how these claims could be measured. The results of the claims are shown in Section 5.1.4 and help to answer the human factors research question.

3.1 Scenario

The design scenario gives a short preview of what problem should be solved by the system design, who are affected by the solution, in which way the system tries to solve the problem, and how people will use the system. The following design scenario is partially based on an existing TRADR-scenario, which is "[a] fixed human-robot team with 1 UGV and 1 UAV gradually builds up situations awareness of a *dynamic* disaster site over multiple *synchronous* or *asynchronous* sorties." [2, p. 6]. An addition to the scenario in this research is a victim rescue part.

Design scenario

The design scenario describes what the desired situation would look like when the team leader uses TTSS.

The USAR team has to get a clear picture of the disaster site. The team leader has to coordinate an initial exploration task. He selects an exploration task for a UAV from a process list. After entering the instructions for the UAV operator, the team leader opens the process model of the exploration process and monitors the situation. He sees the progress in the workflow. He opts for another area that needs exploration and starts a process for another UAV. Meanwhile, in the process model of the first UAV, a task has turned red and an error report has been listed in the team leaders task list. The team leader reads the report about a malfunctioning camera and reassigns the task to another UAV. He starts a repair process for the broken camera. The workflow model shows the exploration task is ongoing again. The team leader stops the other UAV exploration process, as no other UAVs are available in the resource list. An area that has been explored needs further assessment and the team leader sends in a UGV to explore a building with possible victims. Once he receives pictures of located victims, he instructs the UGV to skip shooting more pictures and to find a safe pathway for rescue and he sends his in-field rescuers to the victims by starting a rescue process.

3.2 Use-cases

Use-cases are more detailed scenarios which contain step-by-step information of how the users interact with the system. The use-cases are specifically designed for implementing them into a workflow-based system, such as TTSS [26]. Five use-cases are presented relevant to the domain. For readability only one of the use-cases is presented here. The other use-cases can be found in Appendix A. Often a use-case contains alternative steps, considering the alternative steps the end-user may take during an experiment. As the experiment performed for this study is a joint evaluation of the support system by experimenter and participant, these steps are not present in the current use-cases. These can however be easily added for further experiments.

UC1: Perform exploration process

Actors:	Team leader, TTSS, UAV, UAV pilot, UAV operator
Circumstance:	No information is known about the area
Precondition:	A UAV-team needs to be available
Post condition:	A UAV has taken pictures and the initial exploration is complete
Method:	Commands via TTSS, walkie-talkies
Steps:	1. The team leader looks at the map for the location of exploration. 2. The team leader starts the initial exploration process from the 'My tasks' list. 3. The team leader fills in the coordinates and resource and presses start. 4. TTSS starts the workflow of the exploration process. 5. The UAV team is assigned to the exploration process and executes it. 6. The team leader monitors the progress of the exploration process.

3.3 Requirements

The requirements for TTSS have to be set in a clear and unambiguous manner. These are obtained via the scenarios provided by members of the TRADR-project. The requirements description incorporates the OPD-principles of Johnson [12]. The requirements for the system are put in Table 3.1 for an overview. Claims based on these requirements are discussed in the next section. Requirements R01 until R07 are necessary to ensure observability. R03, R05 until R07 are required for predictability and lastly R08 and R09 are the directability requirements. R10 is a separate requirement for the system, to increase the fidelity of the system and in general test the technical capabilities of TTSS. When taking the workflow technology into account from Figure 1.1, the requirements are necessary to make it possible to change plans in real-time.

3.4 Claims

Claims describe the hypotheses which need to be verified by using the TTSS. The requirements from the previous section support these claims. The claims help to evaluate the functionalities of TTSS and give a clear picture what the system is capable of. These claims are based on the

Code	Requirement	Claim
R01	The system should provide a comprehensive overview of all ongoing processes.	C1, C4
R02	The system should clearly show the progress of the processes.	C1, C2, C3, C4
R03	The system should distinguish clearly between current/completed/future tasks.	C1, C2, C3
R04	The system should clearly show the available team members.	C1, C3
R05	The system should indicate what tasks each team member (should) perform.	C1, C2, C3
R06	The system should send notifications of important changes.	C1, C2, C3, C4
R07	The system should provide an overview of all current tasks to perform.	C1, C4, C5
R08	The system should enable changing the ongoing processes.	C2, C3, C5
R09	The system should make it easy to re-assign tasks to different team members.	C2, C3, C5
R10	The system should simulate USAR-like events, such as earthquakes.	C5

Table 3.1: Requirements table, containing the requirements for TTSS. The first column shows the identifier of the requirement, the second the description and the third column shows the claims each requirement supports. These claims are displayed in Table 3.2.

OPD-principles, as described by Johnson. The following sections elaborate upon these claims by considering the pros and cons of each claim and at least one measure for verifying it. Two measures, stress-levels and level of control can be measured with cognitive task load (CTL) methods [39]. A paper concerning a dynamic task allocation model for human-robot teams has applied CTL successfully in a similar fashion [40]. In Table 3.2 an overview is provided of all claims and their corresponding requirements.

Code	Claim	Requirement
C01	The system enables the team leader to have better situation awareness, i.e. a better overview of the situation and the team members.	R01, R02, R03, R04, R05, R06, R07
C02	The system helps the team leader anticipate future problems in the mission better.	R01, R02, R03, R05, R06, R08, R09
C03	The system gives the possibilities to the team leader to respond to immediate problems better with the help of workflow patterns and online resource re-allocation.	R02, R03, R04, R05, R06, R08, R09
C04	The system reduces the overall cognitive load of the team leader.	R01, R02, R06, R07
C05	The system captures the means for low-to-medium fidelity realistic situations.	R07, R08, R09, R10

Table 3.2: Claims table. Look in Table 3.1 for the supporting requirements.

Claim 1

The first claim suggests an increased sense of observability, thus that the workflows of the processes show clearly what each resource is doing. The system in overall should make it easier for the team leader to keep an overview, i.e. lower stress levels.

Claim:	The system enables the team leader having better situation awareness, i.e. a better overview of the situation and the team members.
Measure:	Stress-levels, time spent on observing, number of detected problems
Benefits:	Less cognitive load for the team leader.
	Easier to get the full picture of the situation.
	The team leader is aware of problems immediately.
Downsides:	Not all information is accessible at once.
	Mental model might work faster than the system.

Claim 2

The second claim is based on workflows showing future tasks of the processes and anticipating likely problems, visualizing future processes explicitly to relief cognitive load of the team leader. The duration of the planning phase of the team leader should be reduced.

Claim:	The system helps the team leader anticipate future problems in the mission better.
Measure:	Time to plan
Benefits:	Future problems can be detected by looking at the workflow models.
	Less problems can occur as planning can be adapted to future and current information.
Downsides:	Team leader might get overloaded to think about future instead of current situation.

Claim 3

The third claim supports the last of the three principles, namely directability. Starting and stopping processes, skipping tasks and such are directive requirements supporting this claim. As giving directives is implemented in the same framework as observing the team, alternating between the two should be relatively easy. The team leader should attend problems faster.

Claim:	The system provides the means to the team leader to respond to immediate problems better with the help of workflow patterns and online resource-reallocation.
Measure:	Time to respond to problems
Benefits:	The means to direct the mission are easy to use. Less communication needs to take place.
Downsides:	The team leader has less control over which problems to attend. The workflow patterns can be hard to interpret mentally.

Claim 4

According to the fourth claim, the team leader should be less burdened with dividing attention synchronously over multiple processes and TTSS should guide the team leader in where to make a decision, such as generating error reports or visualize dysfunctional resources. The team leader would have less stress, yet remains in control over the coordination.

Claim:	The team leader will have a lower cognitive load.
Measure:	Stress-levels, level of control
Benefits:	The team leader can remain focused. The team leader will be less exhausted over time.
Downsides:	The team leader might take less responsibility for the mission. The team leader might rely too much on the support of the system. The team leader could spend more time comprehending how the system works instead of decision making in critical situations.

Claim 5

The last claim involves both research questions and is required for testing whether the workflows can realistically represent the processes of an USAR team and if a support system such as TTSS can be used for faster and more efficient simulation of USAR scenarios, as well as recording data in real-time. Based on a questionnaire (Appendix D) it can be established how end-users experience the support system. The claim distinguishes itself from the other four claims by focusing on the technical aspects of TTSS and in general test the capabilities of a workflow-based system and the incorporated workflow patterns.

Claim: The team leader can deal with low-to-medium fidelity realistic situations.

Measure: User experience questionnaire

Benefits: The workflow models map realistically to the protocols of USAR teams.

End-users can comprehend the structure of workflows and even model processes themselves.

Downsides: The problems are yet not very complex and realistic in the current system.

No complete integration of the support system within the TRADR-project yet.

Chapter 4

Prototype

The current chapter discusses in depth the design of the support system (TTSS) and is based on the analysis in the previous chapter. This chapter is involved with answering the technical research question. TTSS is specifically designed to support the team leader in coordinating his or her USAR team. The system supports the team leader based on the OPDz-principles and adheres to the requirements provided in the previous chapter. The support system includes two new concepts in the domain of USAR. First, it is a prototype support system based on real-time executable workflows. Second, the addition of workflow patterns makes the workflow models suitable for dealing with dynamic situations. The first section describes the conceptualization of the system and the extension of flexible workflow patterns to the ST2 tool. The second section describes the implementation of the workflow models in detail and gives an overview of the interface capabilities.

4.1 Conceptualization

The conceptual idea of TTSS consists of merging ideas from two domains, CWA and flexible workflow technology. CWA is the basis for modeling the USAR processes in workflows. Afterwards, flexible workflow patterns need to be selected that are deemed feasible and efficient for modeling task distribution in a dynamic environment.

4.1.1 Abstraction Hierarchy

In order to create suitable workflow models, an abstraction hierarchy is created according to CWA. The TRADR scenario and USAR protocols are used as input for the AH, which is shown in Figure 4.1. For the construction of workflow models, only the bottom three levels have been implemented. The top two levels are not directly implementable yet. Fortunately, most of the reasoning of the team leader occurs on the bottom three levels [21]. Four different functions are modeled in TTSS, namely reconnaissance, rescue and removal, maintenance and monitoring. Only the first and third has also been described in the provided TRADR scenario [2]. The other two are based on existing USAR protocols[1]. Six processes support the functions. Finally there are twelve physical objects that fulfill the role as resource in the processes. Note that these resources could be either team members or equipment. Ideally, there are always multiple processes linked to a function. The same holds for the relation between physical objects and processes, where multiple objects can fulfill the same resource role in a process. This way, the system is more robust and can execute functions and processes more flexibly. For example in the reconnaissance function it is possible to choose between sending in a UAV or UGV for initial exploration. Depending on the circumstances, e.g. a broken robot, different processes can be started to fulfill the same function. More details on the functions follow in Section 4.2.1.

4.1.2 Workflow Patterns

From Table 2.4 a selection is made of four different workflow patterns. The patterns are selected based on the requirements given in the previous chapter (Section 3.3). These patterns are feasible to implement in ST2, on which TTSS is based, and provide the required flexibility for USAR. A similar approach for selecting the appropriate patterns has been used in a healthcare study, where workflow patterns were used for flexible treatments [37].

The first pattern is **skip a task**. The protocols of USAR describe many tasks which can be executed during processes, yet some of them are more pressing than others. In an emergency, the team leader might decide to prioritize one task and skip another when running parallel tasks (use-case in Appendix A.5).

The second pattern is **on-the-fly**. Whenever a resource is down and cannot perform a necessary task, a new resource should be assigned to that task. This should happen in real-time. Resource re-allocation in real-time is necessary to deal with situations quickly.

The third pattern is **starting/stopping processes**. A UGV might be en route to a building for an assessment. Meanwhile, a part of a building collapses and a new exploration of the disaster site is necessary. The assessment the UGV is currently performing is stopped and a new exploration task is initialized.

The last pattern is **momentary change**. Whenever a sudden event happens, such as a resource that is incapacitated or a hazardous material that is discovered (use-case in Appendix A.4), the situation changes. The team leader should be notified immediately of the new situation and the workflows should incorporate the new information about the team status and environment.

[1] https://www.fema.gov/urban-search-rescue

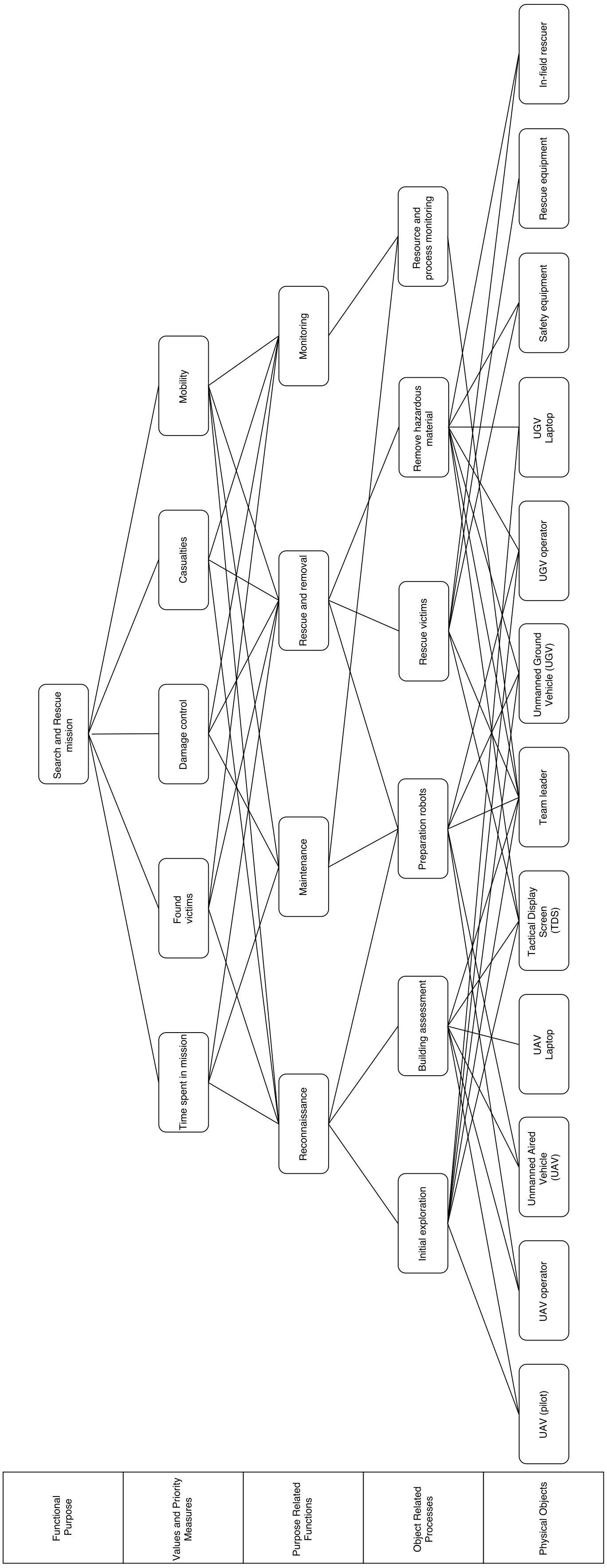


Figure 4.1: The abstraction hierarchy constructed based on scenarios, use-cases and protocols.

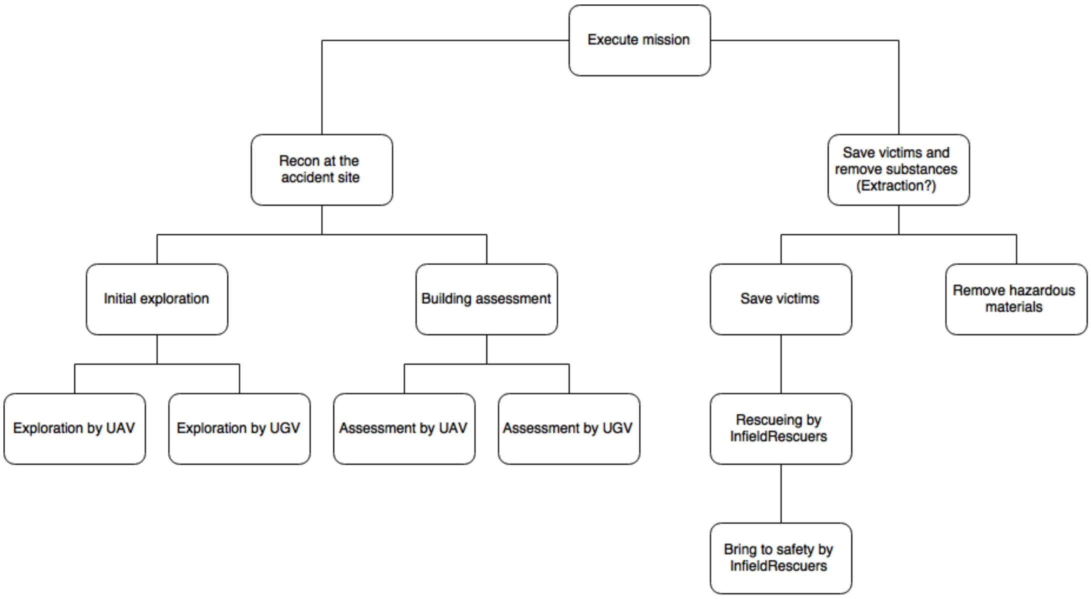


FIGURE 4.2: The mission tree describes how all process models of the mission are related.

4.2 Implementation

TTSS is an extension to the ST2 framework (Section 2.3.2). TTSS has been used to create a low-fidelity prototype of a resilient system for providing support in task distribution in the USAR domain. Specifically, the system helps the team leader to re-allocate tasks to different resources and monitor the team's current status in the ongoing mission. By using a visual representation of the support system, it should increase the comprehensibility of how the system supports the USAR team.

4.2.1 Workflow Models

In total 17 workflow models have been created for TTSS. Their relations are shown in Figure 4.2. The mission tree describes the super/subprocess relations of all processes related to the mission, e.g. Explore by UAV is a subprocess of Explore. These are created based on the TRADR scenario [2]. The workflow models have been created in the Eclipse IDE with the BPMN 2.0 modeler plug-in and are in accordance with standard BPMN styling [33]. All process models can be found in Appendix B.

Four functions have been modeled, which are reconnaissance, rescue and removal, maintenance and monitoring. Below are the descriptions of these functions and which processes and resources can fulfill the functions.

Reconnaissance

The reconnaissance function includes processes as exploration and assessment to fulfill its goal of retrieving an overview. Processes it contains are exploration and assessment processes for the robots. The resources are the robots, the robot operators and the team leader. An example of a process for reconnaissance is shown in Figure 4.3.

The construction of the workflow is discussed in more detail in this paragraph. The example process (Figure 4.3) has a similar structure as the workflow model shown in Figure 2.5. The difference is the symbol in the start and end event. The 'Exploration by UAV' process is a process that receives information from another process and passes information to that same process, namely the process 'Explore' (Figure B.3). Information is defined in basic Java types or customized types made in Java. The start signal contains information such as coordinates of the location, defined as two integers. This information can be used within the 'Exploration by UAV' process by the UAV operator to plan a route and pass that information on to the UAV pilot. Finally, a signal containing information about the exploration mission is sent to the 'Explore' process, such as the number of victims seen and other important details. The 'Explore' process is part of the function 'Reconnaissance' (Figure B.2), which interact via a sub/superprocess relation. The reason for not using signals here, is because 'Reconnaisance' is a function, whereas 'Explore' and 'Explore by UAV' are both processes on the same level. The information passing in a sub/superprocess relation is similar to that of signals, except that the superprocess has to call the subprocess, defined as a property.

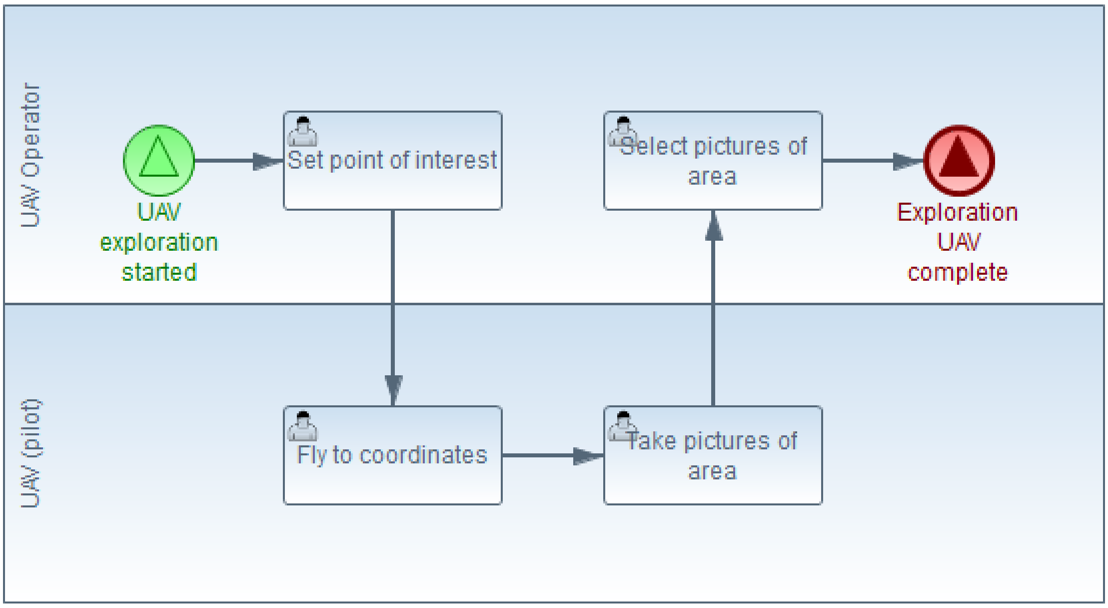


Figure 4.3: The exploration process of a UAV, where both the UAV operator as UAV pilot act as resources.

Rescue and Removal

The extraction function encompasses the processes needed to safely rescue victims and remove dangerous materials. The in-field rescuers and team leader play a part in these processes.

Maintenance

Maintenance is a function where ensuring all team members have the equipment they need. This includes robotic team members, who might require a new battery. Processes belonging to maintenance are returning the robots home or changing the team members for a new sortie, which can be done by the team leader.

Monitoring

Monitoring is a function continuously active to provide the team leader with the means to coordinate the USAR mission. It contains a process view for monitoring processes and giving directives. Also a resource list is available, which the team leader can consult for checking the status of resources.

4.2.2 Interface

The interface of TTSS consists of different windows opened within a web page. The team leader has two screens available for monitoring, namely a list of resources and a list of processes. The resource list shows each available active resource and which task the resource performs (Figure 4.4). The process list shows all current active processes (Figure 4.5). Furthermore, the process list shows the tasks belonging to a process, which resources are performing the tasks and when the tasks were started. The process list also shows three of the four workflow patterns, skip task, start and stop (kill) process and resource re-allocation (re-assign). A task can be skipped by searching for the appropriate task in the process view and pressing 'Skip task' (Figure 4.6a). Starting a process can be done from the bottom-left corner of the process view where you can select a process from a list (Figure 4.6b). Stopping a process is done by finding the process in the process view and press 'Kill process' (Figure 4.6c). A pop-up will confirm the stop. Resource reallocation can be found by searching for the process with the task you want to change resources of and pressing 'Re-assign' (Figure 4.6d). You can then select another resource from the list. Another feature of the process view is to show the status of the processes as live workflows (Figure 4.6e). In blue is marked which task is currently active, grey means a task is completed and white means it is a future task. Whenever a resource becomes incapacitated, a task belonging to that resource will change to red as well, which is part of the last workflow pattern.

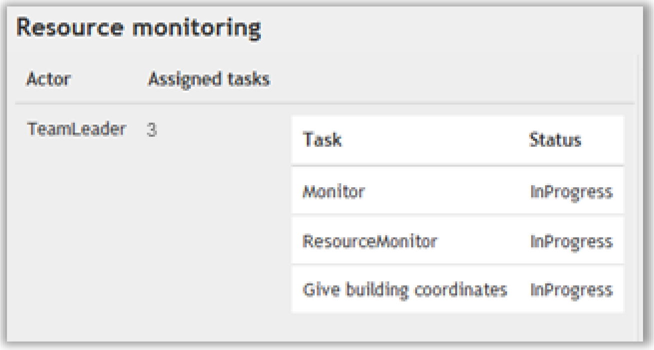


Figure 4.4: A list of resources, showing which resource has which currently assigned task. It also shows if the resource is incapacitated.

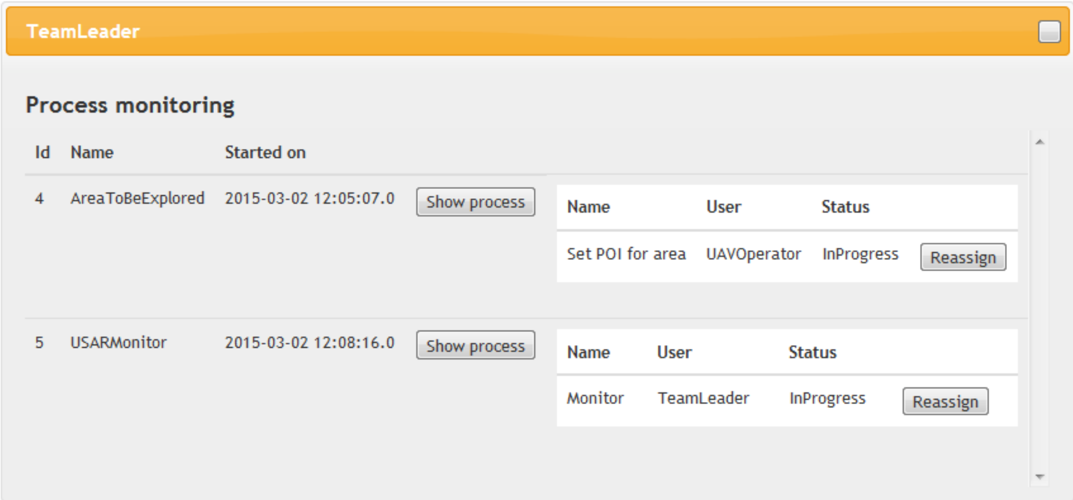
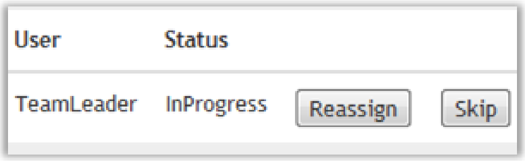
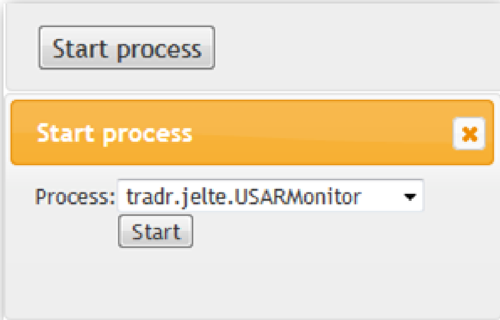


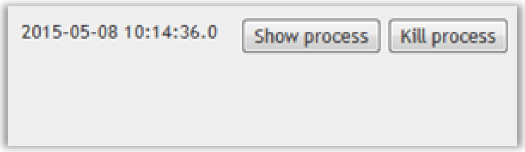
Figure 4.5: The process view contains a list of processes which are currently active, who performs tasks within these processes and what their status is. The process view also gives the ability to influence these processes based on the workflow patterns.



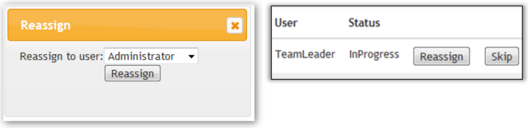
(A) Skip a task.



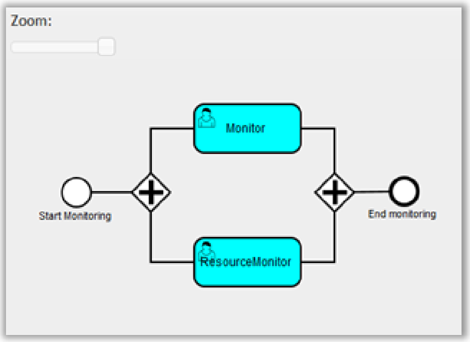
(B) Start a process.

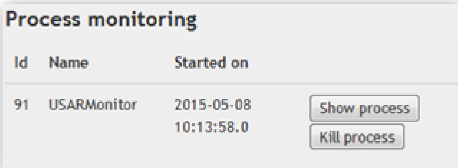


(C) Stop a process.



(D) Re-assign a task.





(E) The process model shows in runtime which task is currently in progress in a process. Hovering over a task shows detailed information about the resource.

FIGURE 4.6: An overview of all of the four selected workflow patterns.

Chapter 5

Evaluation

The current chapter discusses the results related to the two research questions. The first section evaluates the workings of the TRADR Task Support System in the urban search and rescue environment with respect to the human factors research question. The second section focuses on the validating of the support system prototype and answering the technical research question.

5.1 Human Factors

In this section first the experiment is discussed in more detail, such as the artifact describing the devices necessary for the experiment. Afterwards the evaluation method is discussed, containing information about the participant, tasks and set-up. Finally the results with regards to the claims are evaluated.

FIGURE 5.1: The T-JEx site in Phönix West, Dortmund.

5.1.1 Experiment

To evaluate TTSS, the system was tested at the site of a T-JEx[1] in Dortmund. The site provided the necessary context for an USAR scenario (Figure 5.1). Members of the Dortmund Feuerwehr, Vigili del Fuoco Pisa and Gezamenlijke Brandweer Rotterdam were the end-users for the experiment. The TTSS experiment was separate from another larger T-JEx experiment. We aimed to perform the experiment with three to five firefighters. Unfortunately, only one participant was able to help with the TTSS experiment, as all other firemen were occupied with the larger experiment. Therefore only the experience of one end-user is used to reflect on the claims made. The claims can therefore not be confirmed nor rejected, yet still valuable information of the end-user could indicate the plausibility of the claims.

5.1.2 Artefact

TTSS is based on ST2 and workflow models (discussed respectively in Section 2.3.2 and Section 4.2.1). A laptop was provided to work with the TTSS and the participant had to enter information via mouse and keyboard to TTSS. A monitor displayed the use-cases the participant had to perform. The system is designed to support the use-cases in this project, but can easily be extended to include more use-cases. Also an internet connection needs to be available to load the different workflow models and process changes online and in real-time.

5.1.3 Evaluation Method

TTSS is a form of human-computer interaction, for which is it interesting to look at both ends of the interaction what the effects are. Because none of the robots were able to interact with the system and no multiple firefighters were available to act as team members, the Wizard of Oz method is applied. The Wizard of Oz method is useful when no autonomous system is available, yet the participant has to believe the system is already autonomous. The Wizard of Oz is the person mimicking the autonomous system. Via this method the interaction of the participant with the system can be more realistically evaluated. The experimenter had an extra laptop to act as the Wizard.

[1]TRADR Joint Exercise, where all partners of the TRADR project meet up to test their research and products with end-users.

Participant

Unfortunately, only one participant was able to perform the experiment, which was a middle-aged male firefighter of the Gezamenlijke Brandweer. He was healthy and already familiar with the use of robots in urban search and rescue.

Task

The participant had to perform a small tutorial for getting familiar with the system, based on a use-case. Afterwards, the participant was asked for any remarks or questions about the system before continuing the experiment. The experiment consisted of five different use-cases performed jointly with the experimenter. The use-cases involve exploring, assessing a building, dealing with smoke, dealing with an explosion and solving a technical difficulty. These use-cases are presented in Appendix A. The use-cases were created such that all workflow patterns of TTSS were covered. During these use-cases, the participant could ask questions to the experimenter if there was anything he did not understand or needed help with performing certain tasks. After the participant had performed the experiment, he had to complete a questionnaire to evaluate the system. Next to the questions, also a discussion with the participant led to interesting insights, reported in Section 5.1.4.

Materials and Set-up

The set-up of the experiment is shown in Figure 5.2. The experimenter was on the left, performing as a Wizard of Oz. The participant sat at a desk and had two screens next to each other. On the left, the laptop's screen displayed the TTSS and the monitor on the right showed the map and use-cases. The monitor was connected to the experimenters laptop, so that he could cycle through the different use-cases. The laptop was also used to act as the Wizard of Oz.

5.1.4 Results

All claims are separately evaluated using the questionnaire provided to the participant and the notes of the discussion with the participant. An overview of the claims can be seen in Table 3.2.

Claim 1

The claim about observability has not been verified as such. The end-user said that the 'Show process' option was useful, yet he still found the information in one process model to be too detailed. Getting the necessary information went slower than expected. Moreover, he had to

FIGURE 5.2: The set-up of the experiment. The experimenter is shown on the left and the participant is shown on the right.

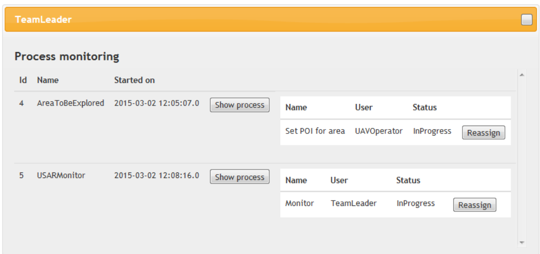


FIGURE 5.3: The interface of TTSS, where a process can be re-assigned to another resource.

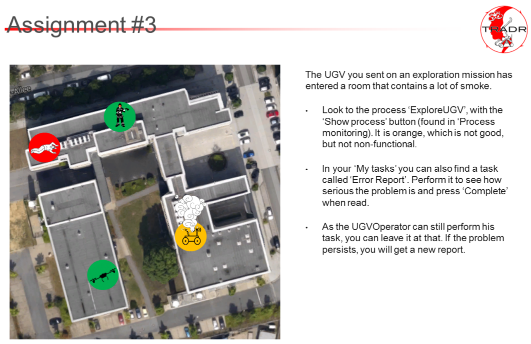


FIGURE 5.4: Scenario + map, which were controlled by the experimenter.

monitor several process models, which was too overwhelming[2]. Thus there is yet no benefit of lower cognitive load. He said that checking the processes is only interesting when processes take longer than a given deadline. After the deadline, the team leader should get information about why it is taking longer and whether he has to readjust the current plan. He considered the coloring of tasks a useful indication in the process model to know some process will exceed the deadline. He felt that the pop-ups showing error reports were indeed attending him better to the problems at hand. However, these reports are silently added to the tasklist of the team leader and have to be clicked on to view more information, making them easier to miss.

Claim 2

For the second claim concerning predictability not much evidence is found. When asking the end-user whether he found the process models useful for looking into future problems, he responded that he does not want to monitor processes for such problems. Again the only interesting part

[2]A side note is that the user interface did not lend itself for quickly switching between process models, which also increased the time to monitor the correct process. Yet this was no part of the evaluation, because it was outside the scope of the experiment.

was when a process has exceeded its deadline. He preferred to draw his attention to immediate problems. No further feedback was given on future problems, it only relied on the team leader monitoring a process and viewing the impact of a task on the entire process. There has not been a reduced planning time ready.

Claim 3

The third claim about directability actually made sense. The end-user was very positive about the directability options in ST2, by making use of the workflow patterns. He said he had no trouble understanding these patterns and said they were much like the actions performed in conventional USAR missions. Especially the easiness of the system to immediately stop a process and start a new one is similar to the current conventions. All options, ‘Start process’, ‘Stop process’, ‘Skip task’ and ‘Re-assignment’ were discovered quickly. The time to respond to problems was fairly fast compared to the conventional approaches. The only downside was that the options were ambiguously positioned, where the difference between changing a process (e.g. ‘End process’) or changing a task (’e.g. ‘Skip task’) was unclear. With some of the use-cases presented to the end-user, he would have followed another course of action than presented in the scenario to solve the use-case. The system did provide the means to deviate from the scenario-based prescribed actions to another team leaders approach of action. This flexibility of directing tasks and processes as the current team leader sees fit, makes this system more universally applicable. A last point to make about the claim is that the team leader does not want to fix every problem, for example the low battery problem of the UGV. The end-user said that problems such as these are mostly solved by the robot operators themselves.

Claim 4

There was no reduced cognitive load noteworthy, mostly because of the focus on monitoring processes. The many clicks and inputs of the team leader were considered too time-consuming. The teamleader felt overwhelmed by the many processes to monitor and the information he had to give. Tasks and processes were modeled too detailed (as mentioned in the evaluation of the first claim). Therefore, the system has not provided any benefits for the lower cognitive load. The possibility of taking less responsibility is however not the case, as the team leader was aware of his responsibility the entire time and did not feel like the system was taking control.

Claim 5

The last claim has been supported the most by the arguments of the end-user. The processes in the workflow model map exactly to the procedures used during USAR missions and the end-user found them easy to interpret. There is a clear description of those protocols, making it fairly easy to model these in workflow models. The mission tree (Figure 4.2) presented to the end-user was correctly structured, though he interpreted it as being similar to a workflow. His interpretation was that an assessment process could be executed before even initial exploration. After explaining the difference between the AH and the workflow models, he would have still liked an overview in workflow-form of all possible processes. The system is still abstract and has much to gain on making the simulation more immersive according to the end-user. These are mostly improvements on the UI side, such as adding a map for selecting locations, having real-time positions of the assets and better sized and organized windows. Despite the difficulties of the interface, the end-user considered it plausible to have workflow-technology applied in the field. Other improvements for the support system would be to leave details out of some processes. The main reason given was that the team leader needs an overview of the entire function more often and should not be bothered with monitoring all separate processes.

5.2 Technology

The evaluation of TTSS is based on an interview with the end-user how he liked the prototype support system itself. First the workflow patterns are discussed and afterwards the advantages and disadvantages of TTSS as a prototype support system is discussed.

5.2.1 Workflow Patterns

In general, the participant said the workflow patterns corresponded to the commands given in conventional USAR teams. For each of the patterns, an evaluation of the end-user is given.

Skip task

The workflow pattern of skipping a task is relevant, though the end-user said it was not entirely clearly positioned on the interface. Simply said, the prioritization provided by this pattern is often what is decided by the team leader.

Start/stop process

A note on stopping a process is that the end-user would have preferred an 'on hold' process. He said that during the use-case of the explosion (Appendix A.4), he wanted the rescue process to continue, just wait for the robots to find another route and then continue with the rescue process. Similar to the 'skip task' pattern, the 'start/stop process' pattern is part of the prioritization that is decided by the team leader.

On-the-fly

Re-assigning a task to another resource was easy and efficient. However, the end user noted that it was unclear whether he changed the resource for an entire process or just the task. He was curious about how the resource was notified of the change. This change is reported similiary as with the team leader, where a task report appears in the 'My tasks' windows of TTSS.

Momentary change

The error reports generated by the workflow system were clear, but took too many clicks to view. The red coloring and status updates of resources did help locate problems, though the end-user found them fairly redundant. He said he did not want to actively monitor resources all the time, because his main job would be to make decisions. Gathering information that is necessary for future decisions is a secondary role for the team leader, because often present problems are more pressing.

5.2.2 Advantages

A major advantage of TTSS is the easiness of modeling workflows. The end-users can create workflows themselves with little workflow-knowledge, because of the visual modeling language. Another advantage is the way the executed workflow model – the plans – adapt to the information given by the team leader and environment. The changes that the team leader needs to make to the workflows with the patterns is sufficient for the scenario provided for the TRADR-project.

5.2.3 Disadvantages

The interface requires a large improvement, as too many screens clutter the usability. The addition of a map on which the resources are visible, as well as the environmental events, should decrease the execution time of the tasks for the team leader. Another disadvantage that is not immediately clear is which process is a top and which is a bottom process, as all processes are presented on the same level in the process view. This makes it harder to reason about decisions compared to other EIDs [6].

Chapter 6

Conclusion

The main issue of this research project was if it was possible to help a USAR team by supporting the task distribution.

- How can a process-aware information system support identifying and resolving sudden issues in the task execution of an urban search and rescue team?

In order to answer the main question, two subquestions were put forward, which are answered separately in short. Afterwards we conclude with an answer to the main research question.

- How can a support system for human-robot USAR missions be specified and evaluated effectively?

The sCE-method has helped to formulate clear requirements and claims to test this research question. The evaluation itself had a drawback with only one participant, though still some information has helped to evaluate the plausibility of the claims. For example directability has proven itself being the most useful for the end-user in coordinating his team. The same method can be repeated, though more participants are required and more quantitative data should be gathered to measure performance of the system effectively.

- How can workflow systems provide a more adaptive task distribution among humans and robots which is needed to cope with the unpredictable nature of USAR missions?

Workflow technology combined with flexible workflow patterns has proven to be effectively useful. The main advantages were the visual aspect of workflows, guaranteeing comprehensibility. In the background, the formal approach to modeling the processes has made it less ambiguous than informal methods. The workflow patterns selected were also the most required and used ones in conventional task distribution in USAR, especially if the directability principle is taken into account. The model provided in Figure 1.1 is a solid basis for adding flexibility in an USAR support system.

To answer the main question, with flexible workflow technology it is possible to create a task support system, based on a clear process model and workflow patterns. We showed that the process model can adapt to changes in the environment and contained the necessary requirements for the end-user to act as team leader in a prototype set-up by incorporating the OPD-principles.

6.1 Discussion

The abstraction hierarchy in Figure 4.1 can be extended to include other parties involved with the rescue for a more complete picture, such as municipalities and medical support. The implemented workflow models are only a part of the constructed AH within USAR. The focus was on reconnaissance and structural inspection, because these are processes that can currently be executed by robots. Future improvements on the robots could render them suitable for other processes as well. Moreover, implementing more workflow models and testing them simultaneously would increase the representativeness of TTSS. This would make the workflow models difficult to comprehend. To decrease the complexity of many workflows, some techniques have been proposed already to ensure readability [41].

More related to the TTSS improvements, distributed control of the mission among the team members might prove to be more effective than a hierarchical structure. In the use-case about the battery replacement for a robot (Appendix A.5), it would be desired that the robot operator can take some decisions himself. As robots increase in their autonomous behaviour during USAR, they could also suggest resource allocations more efficiently. Adding the possibility to put constraints (e.g. battery life, fatigue) on resources, the system could automatically calculate the most efficient task distribution and give advice to the team leader. Enabling resource constraints, based on their capabilities, has been performed successfully already. Awad et al. have used OCL (Object Constraint Language) to model resources' capabilities and facilitate better resource re-allocation [42]. It is even possible to extend the constraints to a more complex level, such as how decisions are made, with hADL (human Architecture Description Language) [43].

The experimental design for answering the human-factor research question was a pilot study and with only one end-user as domain expert, no statistically sound conclusions can be drawn and only few quantitative data were obtained (Appendix C). It was also inconvenient to be experimenter and Wizard-of-Oz simultaneously, leading to undesired waiting times for the end-user during the experiment. Adding automation for simulating the environment should be incorporated in future studies. A follow-up experiment with more users could yield good quantitative data and can be analysed with a cognitive task load (CTL) method to either support or rebut the claims put forward in this research project [39].

6.2 Future Work

Some aspects of the human-robot teaming have not been fully developed during this stage of the TRADR project, yet could yield interesting results, such as a robot operating (semi) autonomously. These robots can interact with the same workflow models as the human team members in TTSS, but need a separate interface for interacting with the system. There is already the possibility within TTSS to implement virtual agents [14]. This could be extended to the USAR robots as actual agents.

The ecological interface design should be improved as well. Though the display is suitable for a versatile number of wearable and non-wearable devices, depending on the role or expertise of the team member, the interface requires different elements. These interfaces require persistent geo-localized information, for which for example TrexCOP was developed [44]. Another study has shown that indeed some individual differences are present when using an EID [45]. The cognitive style, which refers to as the way someone acquires and processes information, is inherent to a person and the interface should be adaptable to one's preference. For mobile devices specifically, a navigational metaphor has been put forward as a more intuitive design [46]. An evaluation of different EID approaches to mobile devices has been performed as well [47]. A second issue of the EID is that it does not take into account falsely obtained information, called 'sensor noise'. The USAR domain contains often much noise, requiring that the display should be enhanced to deal with noisy conditions as well [48]. A final addition to a better EID would be to create the interface multimodally interactive, such as adding different sounds for the team members for different types of notifications [49].

The current TTSS has been tested in an isolated environment and has not been integrated within the larger TRADR experiments. Integration within the TRADR system could be possible by using the ontology provided by the TRADR-project for the creation of process models and interfaces for all team members. Even more so, data of field-experiments of the TRADR-project and real USAR missions together with the ontology could form a basis for process mining. According to Van der Aalst, '[t]he idea of process mining is to discover, monitor and improve real processes (i.e. not assumed processes) by extracting knowledge from event logs readily available in today's systems.' [50]. Simply said, whereas workflow models are often created from a concept top-down, process mining works bottom-up to build the process models from the event logs. Applying process mining to the workflow models could increase their robustness and increase the degree of adjustment to many different situations.

Most BPMN modeling tools are sufficient for creating simple processes, yet it is harder to design workflow models with all possible workflow patterns. Most BPMN tools do only support 'Flexibility by Design' and 'Flexibility by Deviation' workflow patterns. Because there is an increase in the need for dynamic (business) modeling, more extensive modelers will become available, extending for example the BPMN tools with more options to implement the workflow patterns, especially those for 'Flexibility by Change'.

Though the workflow patterns selected in this research project seem sufficient for creating a support system for USAR, some other patterns in Table 2.4 could be looked into for improvements on the system. Future TTSS development could support the use of more workflow patterns. Especially more 'Flexibility by Change' patterns should be implemented and tested, because these are more closely related to immediate changes in the environment.

Only the tip of the iceberg has been revealed of applying dynamic workflows in the context of urban search and rescue. Much research in the field of dynamic workflows and USAR lies ahead, steadily bridging the gap between technology and humans in USAR.

Appendix A

Use-cases

A.1 UC1: Perform exploration process

Actors: Team leader, TTSS, UAV, UAV pilot, UAV operator

Circumstance: No information is known about the area

Precondition: A UAV-team needs to be available

Post condition: A UAV has taken pictures and the initial exploration is complete

Method: Commands via TTSS, walkie-talkies

Steps:

1. The team leader looks at the map for the location of exploration.
2. The team leader starts the initial exploration process from the 'My tasks' list.
3. The team leader fills in the coordinates and resource and presses start.
4. TTSS starts the workflow of the exploration process.
5. The UAV team is assigned to the exploration process and executes it.
6. The team leader monitors the progress of the exploration process.

A.2 UC2: Perform assessment process

Actors: Team leader, TTSS, UGV, UGV operator

Circumstance: No information is known about the building

Precondition: UGV and UGV operator need to be available, initial exploration is already performed

Post condition: UGV has taken a sample and initial exploration is complete

Method: Commands via TTSS, walkie-talkies

Steps:

1. The team leader looks at the map for access points of the building.
2. The team leader fills in the coordinates and resource and presses start.
3. TTSS starts the workflow of the assessment process.
4. The UGV operator is assigned to the assessment process and executes it.
5. The team leader monitors the progress of the assessment process.

A.3 UC3: Dealing with smoke

Actors: Team leader, TTSS, UGV, UGV operator

Circumstance: The UGV is assessing a building with smoke

Precondition: Smoke obscures the vision of the UGV

Post condition: The smoke is dealt with

Method: Walkie-talkies, notification of problems, indicate problem in the workflow

Steps:

1. Smoke has developed in the building the UGV is assessing.
2. The team leader views the error report from the task list and no immediate danger is found.
3. The team leader views the task in the workflow with the process model.
4. The team leader considers the necessity of extinguishing the source of the smoke.
5. The team leader communicates with the UGV operator that he must assess other parts of the building first.
6. The UGV operator continues assessing the other parts of the building and returns later to the current room.
7. The team leader closes the report and workflow model of the smoke report.

A.4 UC4: Dealing with explosion

Actors: Team leader, TTSS, UAV, UAV pilot, UAV operator, in-field rescuer, victims

Circumstance: A gas explosion has caused a partial collapse and blocks the access point

Precondition: In-field rescuers are on their way to save victims, victims are present, a UAV-team is available

Post condition: In-field rescuers are safe and a new access point is found

Method: Walkie-talkies, notifications of problem

Steps:

1. The team leader reads an error report in TTSS.
2. The team leader looks at the process model of the save victims process.
3. The team leader communicates with the in-field rescuers about the current situation.
4. The team leader aborts the process for the in-field rescuers.
5. The team leader opens the resource window for available or non-busy resources.
6. The team leader starts a new assessment process for the UAV and notifies the UAV-team.
7. The team leader inserts the coordinates for a new assessment and notes that there is a possible gas leak.
8. The in-field rescuers are now safe outside, awaiting further orders.
9. The UAV is flying to find a new safe access point for getting to the victims.
10. The team leader opens the new assessment process for ensuring fast rescue of the victims.

A.5 UC5: Dealing with robot problem

Actors: Team leader, TTSS, UGV, UGV operator

Circumstance: The UGV has a low batter level

Precondition: TTSS is running, UGV executing parallel tasks

Post condition: UGV can successfully execute one task and return safely

Method: Commands via TTSS, walkie-talkies, notifications of problem

Steps:

1. The team leader gets a notification of the UGV, telling it is not able to complete his tasks.
2. The team leader reads the error report about battery problems and opens the process model.
3. The team leader communicates with the UGV operator about the current parallel tasks.
4. The team leader modifies the model to skip the task of taking samples and only let the UGV complete the task of finding stable access points.
5. The UGV operator continues with completing the single task of finding stable access points.

Appendix B

Workflow models

B.1 Functional purpose

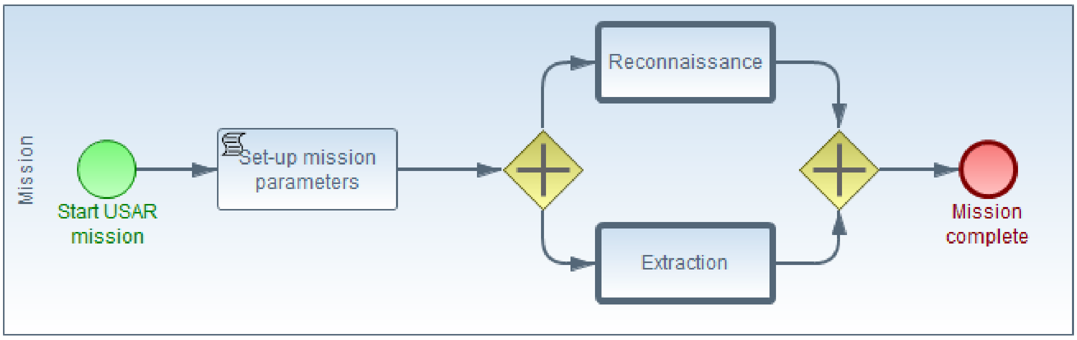


FIGURE B.1: Mission, the top-most process representing the functional purpose of the mission.

B.2 Reconnaissance

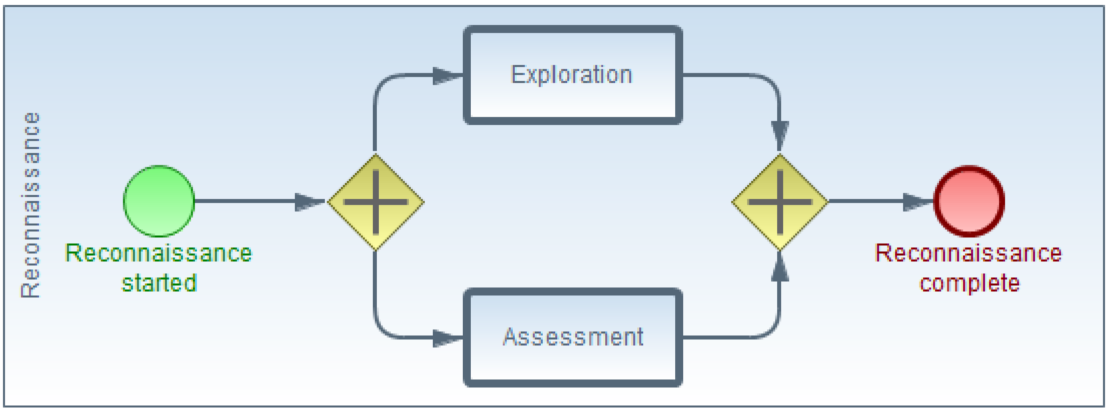


FIGURE B.2: Reconnaissance

B.2.1 Exploration

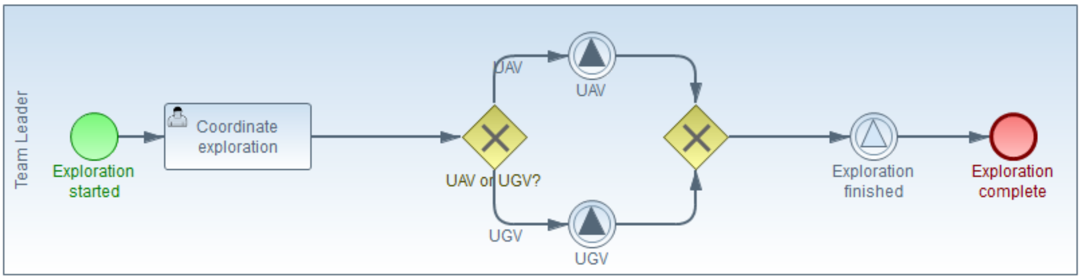


FIGURE B.3: Explore

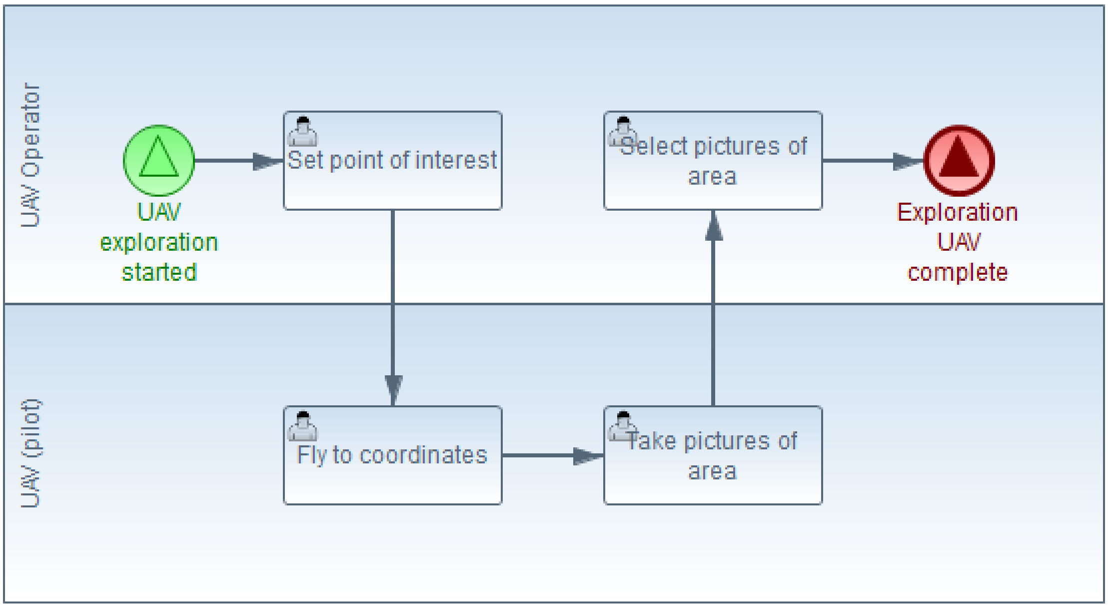


FIGURE B.4: Exploration by UAV

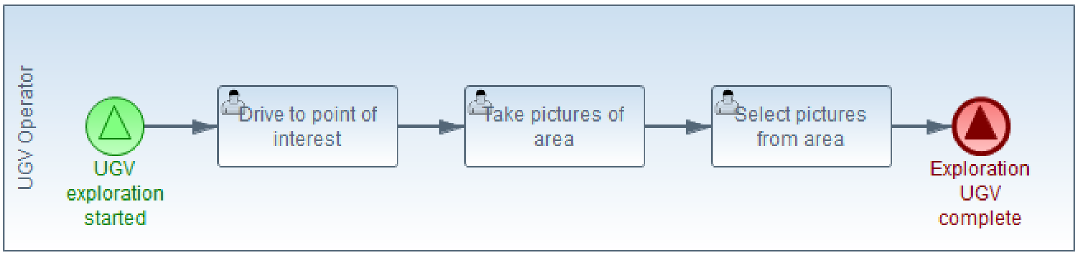


FIGURE B.5: Explore by UGV

B.2.2 Assessment

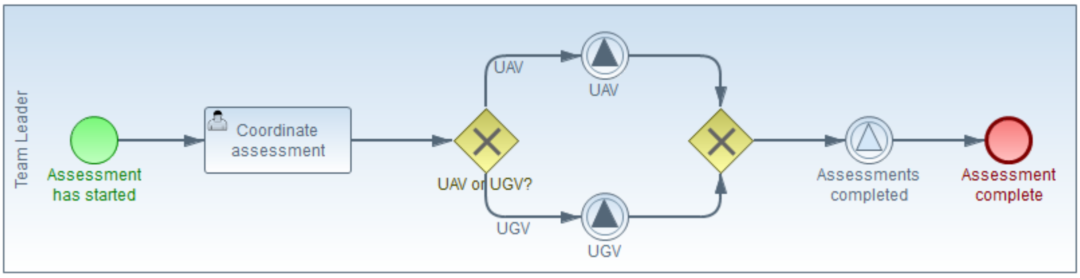


FIGURE B.6: Assessment

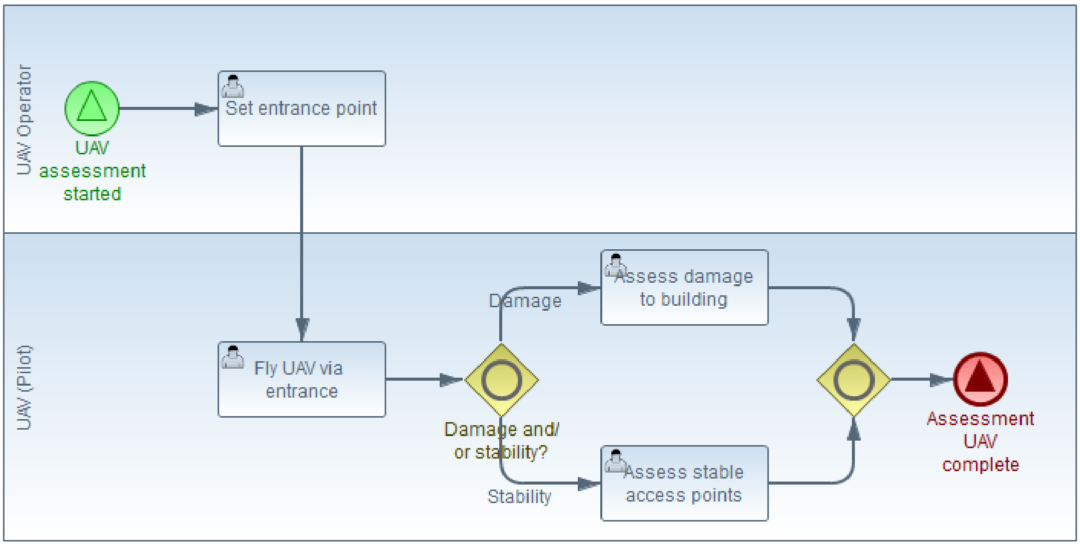


FIGURE B.7: Assessment by UAV

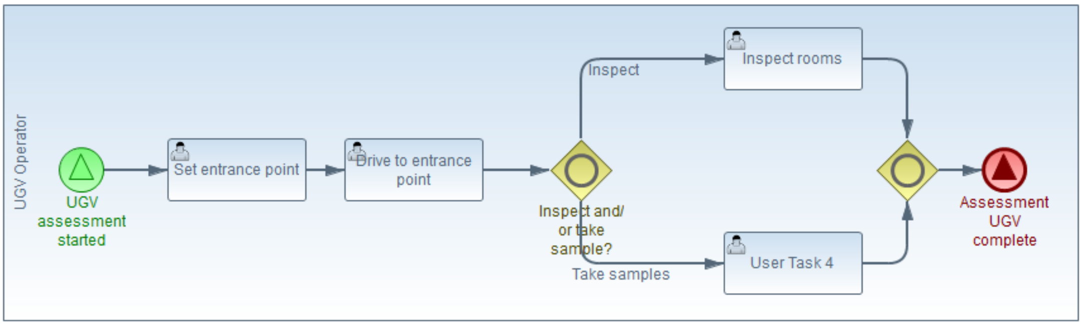


FIGURE B.8: Assessment by UGV

B.3 Rescue and removal

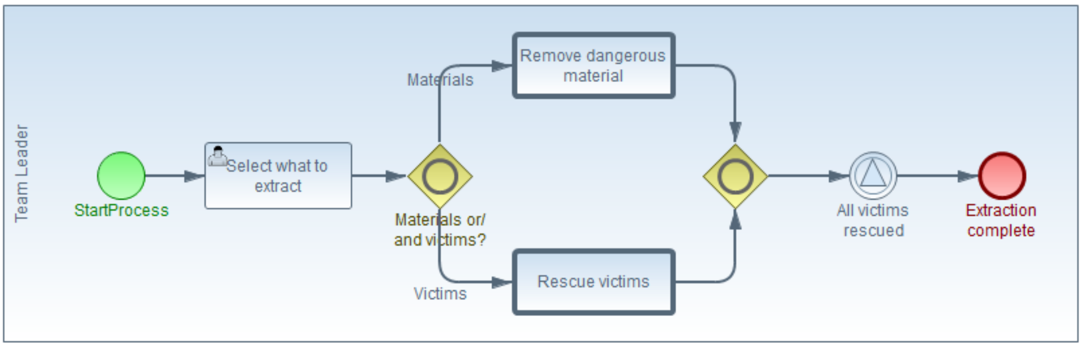


FIGURE B.9: Rescue and removal

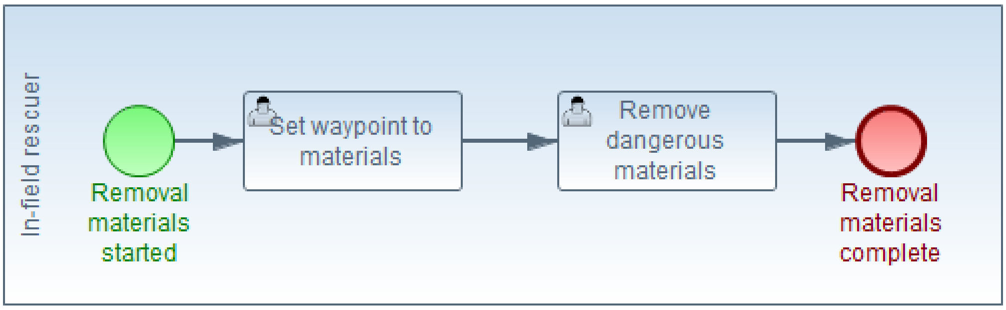


FIGURE B.10: Remove hazardous materials

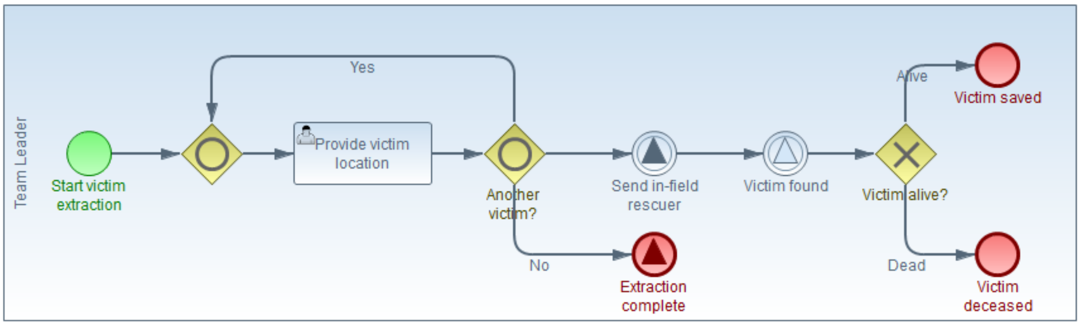


FIGURE B.11: Coordinate victim rescue

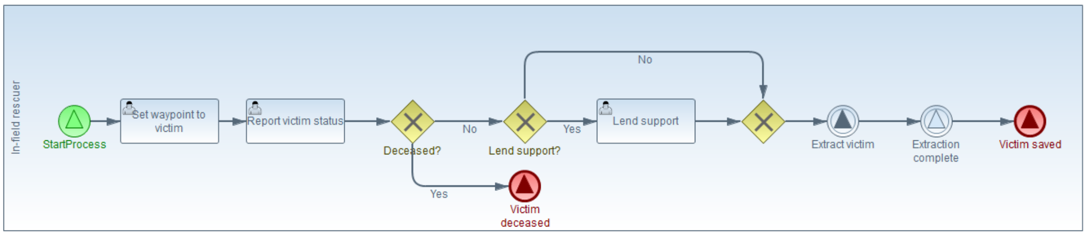


FIGURE B.12: Rescue victims

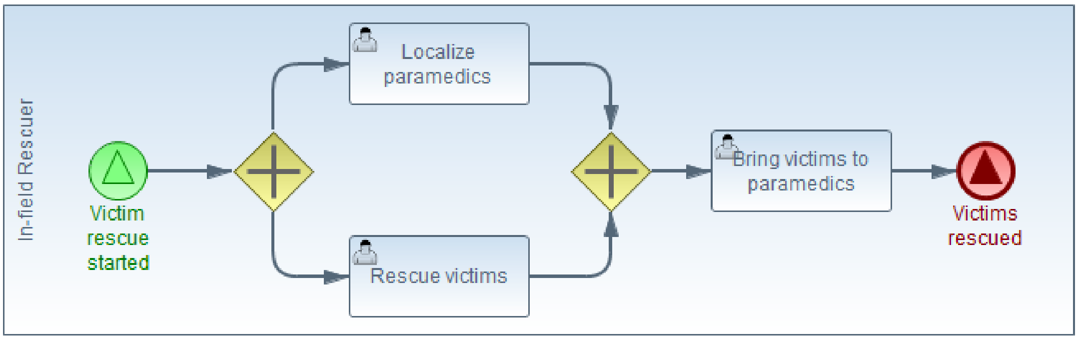


FIGURE B.13: Bring victims to safety

B.4 Maintenance

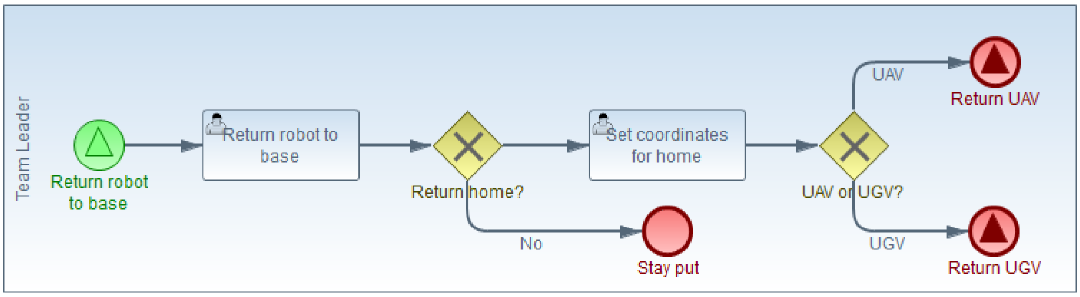


FIGURE B.14: Return a robot home

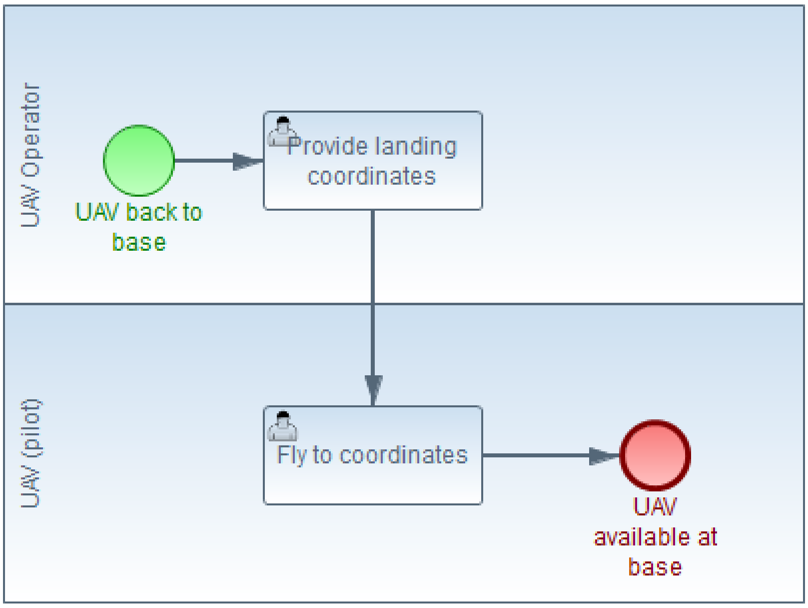


FIGURE B.15: Return a UAV home

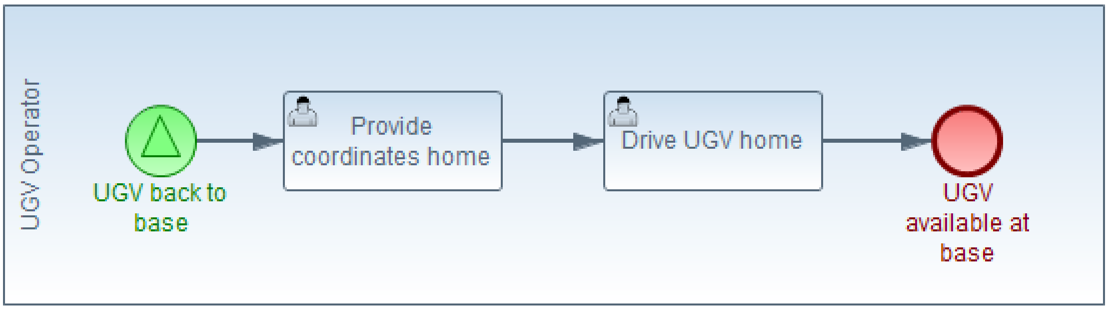


FIGURE B.16: Return a UGV home

B.5 Monitoring

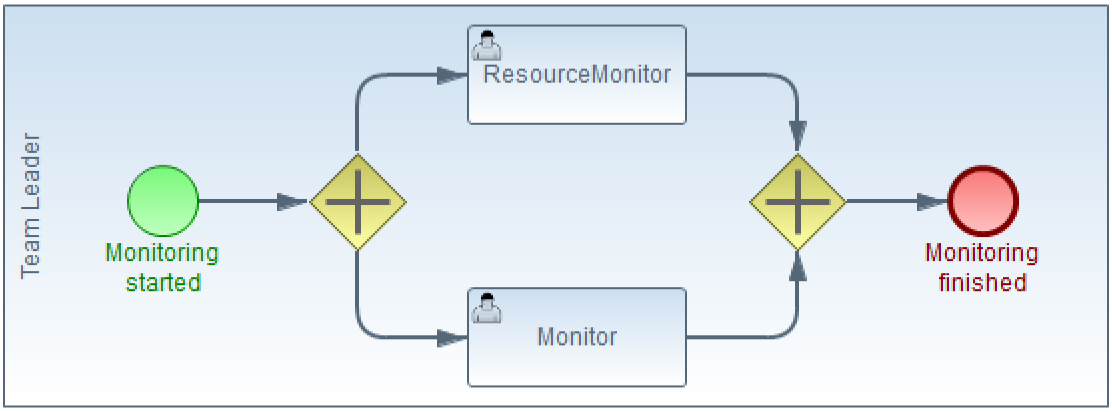


FIGURE B.17: Monitor

Appendix C

Event log experiment

Id	Process name	Started on	Status			
			Task name	Resource	Started on	Status
322	AssessUGV	2015-05-20 13:09:02.0	Set access point	UGVOperator	2015-05-20 13:09:02.0	InProgress
321	AssessUAV	2015-05-20 13:02:22.0	Set entrance for UAV and jobs	UAVOperator	2015-05-20 13:02:22.0	Completed
			Fly UAV via entrance	UAVPilot	2015-05-20 13:02:45.0	Completed
			Assess damage building	UAVPilot	2015-05-20 13:11:07.0	InProgress
			Assess stability access building	UAVPilot	2015-05-20 13:11:07.0	Obsolete
320	ExploreUGV	2015-05-20 12:59:18.0	Set POI and drive to area	UGVOperator	2015-05-20 12:59:18.0	Completed
			Take pictures of area	UGVOperator	2015-05-20 13:00:07.0	Completed
			Select pictures from area	UGVOperator	2015-05-20 13:00:17.0	InProgress
318	SimMain	2015-05-20 12:56:24.0	Start smoke	sim	2015-05-20 12:56:24.0	Completed
			Error Report	TeamLeader	2015-05-20 13:03:17.0	Completed
			End smoke	sim	2015-05-20 13:04:17.0	Completed
			Start explosion	sim	2015-05-20 13:04:34.0	Completed
			Error Report	TeamLeader	2015-05-20 13:04:47.0	Completed
			Kill saving process	sim	2015-05-20 13:09:47.0	Completed
			Start battery level	sim	2015-05-20 13:09:53.0	Completed
			Error Report	TeamLeader	2015-05-20 13:10:01.0	Completed
			Skipped damage task	sim	2015-05-20 13:10:25.0	Completed
319	SaveVictimsIF	2015-05-20 12:56:24.0	Set waypoint to victim location	InfieldRescuer	2015-05-20 12:56:24.0	Completed
			Report status victims	InfieldRescuer	2015-05-20 12:56:55.0	Exited
317	USARMonitor	2015-05-20 12:56:18.0	ResourceMonitor	TeamLeader	2015-05-20 12:56:18.0	InProgress
			Monitor	TeamLeader	2015-05-20 12:56:18.0	InProgress
314	Reconnaissance	2015-05-20 12:56:11.0				
315	Explore	2015-05-20 12:56:11.0	Enter area for exploration	TeamLeader	2015-05-20 12:56:11.0	Completed
316	Assess	2015-05-20 12:56:11.0	Enter building coordinates	TeamLeader	2015-05-20 12:56:11.0	Completed
			Enter building coordinates	TeamLeader	2015-05-20 13:02:22.0	Completed
			Enter building coordinates	TeamLeader	2015-05-20 13:09:02.0	InProgress

Table C.1: Table showing the data of the end user with TTSS.

Appendix D

Questionnaire

Questionnaire ST2

This questionnaire contains questions using the Socio-technical Simulation toolkit, ST2, in the domain of urban search and rescue, specifically regarding the TRADR-project.

About you

1. **Participant ID:** ____________________
2. **Your name:** ____________________
3. **Occupation:** ____________________

ST2

Expectations of ST2

4a. **What did you expect of the tool?**

____

____

4b. **What elements did you miss in the tool?**

____

____

4c. **What elements were surprisingly useful?**

____

____

Usability of ST2

5a. **Do you consider the tool helpful?**
totally not □—□—□—□—□ absolutely

5b. **Would you consider using this tool in the future?**
totally not □—□—□—□—□ absolutely

5c. **Did you feel in control using the tool?**
totally not □—□—□—□—□ absolutely

5d. **Do you think you could give complete instructions?**
totally not □—□—□—□—□ absolutely

5e. **Which of the following monitoring functions are useful?**
□ Resource list □ Processes view □ Show process

5g. **Which of the following directing functions are useful?**
□ Start process □ Kill process □ Skip process □ Re-assigning

5h. **Was it clear to you which resource could perform which task?**
totally not □—□—□—□—□ absolutely

5i. **Could you respond quicker to sudden events?**
totally not □—□—□—□—□ absolutely

Interface of ST2

6a. **How well would the tool be accepted in a search and rescue domain?**
totally not □—□—□—□—□ absolutely

6b. **Did you think the 'My tasks' list is useful?**
annoying □—□—□—□—□ helpful

6b. **Were the pop-ups of tasks annoying or helpful?**
annoying □—□—□—□—□ helpful

Model of ST2

7a. **Was the scenario presented realistic?**
totally not □—□—□—□—□ absolutely

7b. **Do you think tasks are realistically represented?**
totally not □—□—□—□—□ absolutely

7c. **Do you think task re-allocation this way is sufficient?**
totally not □—□—□—□—□ absolutely

General questions

8. **Do you like the tool in general?**
□ Yes □ No □ Not sure

9. **Do you think the tool would make rescue missions more effective?**
□ Yes. □ Probably not. □ Don't know.

10a. **In case you have any further comments, here is some space for writing them down:**

______

______

______

______

Bibliography

[1] Robin R. Murphy, Satoshi Tadokoro, Daniele Nardi, Adam Jacoff, Paolo Fiorini, Howie Choset, and Aydan M Erkmen. Search and Rescue Robotics. *Search*, pages 1151–1173, 2008. doi: 10.1007/978-3-540-30301-5.

[2] Ivana Kruijff-Korbayová, Francis Colas, Mario Gianni, Fiora Pirri, Joachim de Greeff, Koen Hindriks, Mark Neerincx, Petter Ögren, Tomáš Svoboda, and Rainer Worst. TRADR Project: Long-Term Human-Robot Teaming for Robot Assisted Disaster Response. *KI - Künstliche Intelligenz*, page 11, 2015. ISSN 0933-1875. doi: 10.1007/s13218-015-0352-5. URL http://link.springer.com/10.1007/s13218-015-0352-5.

[3] Robin R. Murphy. Human–Robot Interaction in Rescue Robotics. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 34(2):138–153, May 2004. ISSN 1094-6977. doi: 10.1109/TSMCC.2004.826267. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1291662.

[4] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, Patrick Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis. The SHERPA project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments. *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–4, November 2012. doi: 10.1109/SSRR.2012.6523905. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6523905.

[5] J.L. Casper and Robin R. Murphy. Workflow study on human-robot interaction in USAR. *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, 2(May):1997–2003, 2002. doi: 10.1109/ROBOT.2002.1014834. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1014834.

[6] Minh Po Sam. Ecological approach to develop a shared mental model for collaboration at a distance during a crisis situation, 2009.

[7] Jennifer L. Burke, Robin R. Murphy, Michael D. Coovert, and Dawn L. Riddle. Human – Computer Interaction Moonlight in Miami: Field Study of Human-Robot Interaction in

the Context of an Urban Search and Rescue Disaster Response Training Exercise. *Human–Computer Interaction*, 19:85–116, 2004. doi: 10.1080/07370024.2004.9667341.

[8] Mark A. Neerincx and Jasper Lindenberg. Situated cognitive engineering for complex task environments. In *Naturalistic decision making and macrocognition*, chapter 18, pages 373–390. Ashgate Publishing Limited, Hampshire, 1st edition, 2008.

[9] N.J.J.M. Smets, I. Cohen, M.A. Neerincx, W.P. Brinkman, and J. van Diggelen. Improving crew support methods in human-machine teams for long-durations missions. In *63rd International Astronautical Congress 2012*, pages 103–111, January 2012. URL http://www.narcis.nl/publication/RecordID/oai:tudelft.nl:uuid:74caed4a-45aa-4867-8e81-14a0238f522e.

[10] W.M. Post, M. Grootjen, H. Kuiper, M.L.W. Rakhorst-Oudendijk, and J.M.C. Schraagen. Methods for Socio-Technical System Analysis applied to OPV Operational Maintenance. Technical report, TNO, 2013.

[11] Bas Lijnse, Jan Martin Jansen, Ruud Nanne, and Rinus Plasmeijer. Capturing the Netherlands Goast Guard's SAR Workflow with iTasks. In *Proceedings of the 8th International Conference on Information Systems for Crisis Response and Management, ISCRAM*, volume 11, 2011.

[12] Matthew Johnson. *Coactive Design: Designing Support for Interdependence in Human-Robot Teamwork*. PhD thesis, TU Delft, 2014.

[13] Helen Schonenberg, Ronny Mans, Nick Russell, Nataliya Mulyar, and Wil van der Aalst. Process flexibility: A survey of contemporary approaches. In *Lecture Notes in Business Information Processing*, volume 10 LNBIP, pages 16–30, 2008. ISBN 9783540686439. doi: 10.1007/978-3-540-68644-6_2.

[14] Jurriaan Van Diggelen, Wilfried M. Post, and Marleen Rakhorst. Using process-oriented interfaces for solving the automation paradox in highly automated navy vessels, 2014.

[15] Rosemarijn Looije and Mark A. Neerincx. Situated Cognitive Engineering to Support Systematic Study of "Social Features" in HRI, 2012.

[16] Christian Heath and Paul Luff. Collaboration and control: Crisis management and multimedia technology in London Underground Line Control Rooms. *Computer Supported Cooperative Work*, 1(1-2):69–94, 1992. ISSN 09259724. doi: 10.1007/BF00752451.

[17] C.J. Van Aart and A.H.J. Oomes. Real-time organigraphs for collaboration awareness, 2008. URL http://mmi.tudelft.nl/~stijn/collaboration/ISCRAM2008VanAartOomes.pdf.

[18] Mica R. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995. ISSN 00187208. doi: 10.1518/001872095779049543.

[19] Patrick Doherty and Fredrik Heintz. A delegation-based cooperative robotic framework. In *2011 IEEE International Conference on Robotics and Biomimetics*, pages 2955–2962. Ieee, December 2011. ISBN 978-1-4577-2138-0. doi: 10.1109/ROBIO.2011.6181755. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6181755.

[20] Corine H.G. Horsch, Nanja J.J.M. Smets, Mark A. Neerincx, and Raymond H. Cuijpers. Revealing unexpected effects of rescue robots' team-membership in a virtual environment. In *ISCRAM 2013: Proceedings of the 10th International Conference on Information Systems for Crisis Response and Management, Baden-Baden, Germany, 12-15 May 2013*. ISCRAM, 2013.

[21] Jens Rasmussen. The role of hierarchical knowledge representation in decisionmaking and system management. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(2): 234–243, 1985.

[22] Daniel P. Jenkins, Neville A. Stanton, Paul M. Salmon, and Guy. H. Walker. *Cognitive Work Analysis: Coping with Complexity*. Ashgate Publishing Limited, 2009.

[23] Catherine M. Burns and Kim J. Vicente. Model-Based Approaches for Analyzing Cognitive Work: A Comparison of Abstraction Hierarchy, Multilevel Flow Modeling, and Decision Ladder Modeling. *International Journal of Cognitive Ergonomics*, 5(3):357–366, 2001.

[24] Michael E. Janzen and Kim J. Vicente. Attention Allocation within the Abstraction Hierarchy. *International Journal of Human-Computer Studies*, 48(4):521–545, 1998. URL http://apps.webofknowledge.com/full_record.do?product=UA&search_mode=GeneralSearch&qid=15&SID=P1CP4xx2f2j8jOWQu66&page=1&doc=1.

[25] Kim J. Vicente and Jens Rasmussen. Ecological Interface Design: Theoretical Foundations. *Systems, Man and Cybernetics, IEEE Transactions on*, 22(4):589–606, 1992.

[26] Andreas Gehlert, Olha Danylevych, and Dimka Karastoyanova. From requirements to executable processes: A literature study. *Lecture Notes in Business Information Processing*, 43 LNBIP:42–52, 2010. ISSN 18651348. doi: 10.1007/978-3-642-12186-9_5.

[27] Jens Rasmussen and Kim J. Vicente. Coping with human errors through system design: implications for ecological interface design. *International Journal of Man-Machine Studies*, 31(5):517–534, 1989.

[28] Kim J. Vicente. Ecological Interface Design, Progress and Challenges. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 44(1):62–78, 2002.

[29] Patrice Terrier and Jean-Marie Cellier. Depth of processing and design-assessment of ecological interfaces: Task analysis. *Int. J. Human-Computer Studies*, 50(4):287–307, 1999.

[30] J. Kwok and Catherine M. Burns. Usability Evaluation of a Mobile Ecological Interface Design Application for Diabetes Management. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 49(11):1042–1046, September 2005. ISSN 1071-1813. doi: 10.1177/154193120504901109. URL http://pro.sagepub.com/lookup/doi/10.1177/154193120504901109.

[31] Peter Lawrence. *Workflow Handbook*. John Wiley & Sons, Inc., 1997.

[32] Wil van der Aalst and Kees Max van Hee. *Workflow management: models, methods, and systems*. MIT press, 2004. ISBN 0262011891.

[33] Bruce Silver. *BPMN Method and Style*. Cody-Cassidy Press, Aptos, CA, 2nd edition, 2011. ISBN 9780982368114.

[34] Wil van der Aalst, A.H.M. Hofstede, and Mathias Weske. Business Process Management: A Survey. *Springer*, pages 1—-12, 2003.

[35] Dennis Bokermann, Christian Gerth, and Gregor Engels. Use Your Best Device ! Enabling Device Changes at Runtime. pages 357–365, 2014.

[36] Elyes Lamine, Abdel Rahman H Tawil, Rémi Bastide, and Hervé Pingaud. An ontology-driven approach for the management of home healthcare process. In *Enterprise Interoperability VI*, pages 151–161. Springer, 2014.

[37] Hajo A. Reijers, Nick Russell, Simone Van der Geer, and Gertruud A.M. Krekels. Workflow for healthcare: A methodology for realizing flexible medical treatment processes. In *Business Process Management Workshops*, pages 593–604. Springer, 2010.

[38] Rinus Plasmeijer, Bas Lijnse, Steffen Michels, Peter Achten, and Pieter Koopman. Task-oriented programming in a pure functional language. In *Proceedings of the 14th symposium on Principles and Practice of Declarative Programming*, pages 195–206. ACM, 2012.

[39] M.A. Neerincx. Cognitive task load design: model, methods and examples. *Handbook of Cognitive Task Design*, pages 283—-305, 2003.

[40] T.R.A. Giele, T. Mioch, M.A. Neerincx, and J.J.C. Meyer. Dynamic task allocation for human-robot teams. 2015.

[41] Manfred Reichert. Visualizing large business process models: Challenges, techniques, applications. In *Business Process Management Workshops*, pages 725–736. Springer, 2013.

[42] Ahmed Awad, Alexander Grosskopf, Andreas Meyer, and Mathias Weske. Enabling resource assignment constraints in BPMN. *Hasso Plattner Institute, Potsdam*, 2009.

[43] Christoph Dorn, Schahram Dustdar, and Leon J. Osterweil. Specifying flexible human behavior in interaction-intensive process environments. In *Business Process Management*, pages 366–373. Springer, 2014.

[44] Benoit Larochelle, Geert-Jan M. Kruijff, Nanja J.J.M. Smets, and Jurriaan van Diggelen. Experiences with usar mobile interfaces: The need for persistent geo-localized information. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5333–5338. IEEE, 2013.

[45] Gerard L. Torenvliet, Greg A. Jamieson, and Kim J. Vicente. Making the most of ecological interface design: the role of individual differences. *Applied Ergonomics*, 31:395–408, 2000.

[46] Jorge Cardoso, Stefan Jablonski, and Bernhard Volz. A navigation metaphor to support mobile workflow systems. In Niels Lohmann, Minseok Song, and Petia Wohed, editors, *Business Process Management Workshops*, volume 171 of *Lecture Notes in Business Information Processing*, pages 537–548. Springer International Publishing, 2014. ISBN 978-3-319-06256-3. doi: 10.1007/978-3-319-06257-0_43. URL http://dx.doi.org/10.1007/978-3-319-06257-0_43.

[47] Natalia Mazaeva and Ann M. Bisantz. Ecological displays, information integration, and display format an empirical evaluation across multiple small displays. *Journal of Cognitive Engineering and Decision Making*, page 1555343414521424, 2014.

[48] Olivier St-Cyr, Greg A. Jamieson, and Kim J. Vicente. Ecological interface design and sensor noise. *International Journal of Human-Computer Studies*, 71(11):1056–1068, November 2013. ISSN 10715819. doi: 10.1016/j.ijhcs.2013.08.005. URL http://linkinghub.elsevier.com/retrieve/pii/S1071581913001031.

[49] Tobias Hildebrandt. Short paper: Towards enhancing business process monitoring with sonification. In *Business Process Management Workshops*, pages 529–536. Springer, 2014.

[50] Wil Van der Aalst. *Process mining: discovery, conformance and enhancement of business processes*. Springer Science & Business Media, 2011.